Sterling Selling and Fulfillment Foundation

Configuration Deployment Tool Guide

Version 9.1

Sterling Selling and Fulfillment Foundation

Configuration Deployment Tool Guide

Version 9.1

Note

Before using this information and the product it supports, read the information in “Notices” on page 61.

Copyright

This edition applies to the 9.1 Version of IBM Sterling Selling and Fulfillment Foundation and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Chapter 1. Deployment Tools Overview

Deployment Tools Overview

During incremental configurations of IBM Sterling Selling and Fulfillment Foundation, changes are typically developed in a test environment and then they are rolled out into production. Migrating configuration data can be fairly cumbersome and time consuming. Sterling Selling and Fulfillment Foundation provides a Configuration Deployment Tool that enables you to migrate configuration data. This tool ensures data integrity while reducing the system downtime to transfer data and minimizing the effort needed to ensure accuracy.

This tool is designed to migrate data that is modified as part of a normal day-to-day operation. Note that the Configuration Deployment Tool can be used to deploy configuration data that is the result of an upgrade, but it should not be used to perform the data upgrade itself.

The Configuration Deployment Tool (CDT) can be accessed from the Sterling Selling and Fulfillment Foundation Development and Deployment WorkBench (also known as the "WorkBench").

To run the UI for the Sterling Selling and Fulfillment Foundation Configuration Deployment Tool on a UNIX server from a windows client, the UNIX server must have XWindows installed on it.

The CDT provides the following capabilities:

- Transfer complete and partial sets of configuration data or discrete logical portions.
- Transfer data to and from XML files or databases.
- Transform certain data-like IP addresses and port numbers that are different in two environments, depending upon network configuration.
- Generate a report of configuration differences by comparing the two systems.

Chapter 2. CDT Concepts

CDT Source and Target Environments

The Sterling Selling and Fulfillment Foundation Configuration Deployment Tool deploys data from one Sterling Selling and Fulfillment Foundation environment to another. The deployment could occur from development to test environments, from staging to production environments, and so forth. The environment that serves as the point of origin for the data is known as the "source" environment. The destination environment into which data is deployed is defined as the "target" environment. This deployment can be in the form of importing and exporting data to and from databases or XML files.

If you make any changes to the configuration data in the source database, the existing transactions in the CDT may be affected.

Also, for the YFS_TRAN_LOCN_ATTRS table, only the configuration data is copied to the target database. The transaction data columns such as Pend_in_volume, Pend_in_weight, and Freeze on variance are not copied to the target database. The weight and volume are recalculated and updated in the target database.

The following example shows the required format of these XML files.

```
<?xml version="1.0" encoding="UTF-8" ?>
<YFS_ATP_RULESList TableName="YFS_ATP_RULES">
   <AtpRules AccumulationDays="730" AdvanceNotificationTime="0"
   AtpRule="DEFAULT" AtpRuleKey="DEFAULT" AtpRuleName="DEFAULT"
   BackwardConsumptionDays="730" ConsiderPoForAlloc="N" Createprogid=""
   Createts="" Createuserid="SYSTEM" ForwardConsumptionDays="730" Lockid="0"
   MaxInventoryHoldDays="730" Modifyprogid="SYSTEM" Modifyts=""
   Modifyuserid="SYSTEM" OrganizationCode="DEFAULT" PastDueDemandDays="730"
   PastDueSupplyDays="730" ProcessingTime="0" />
</YFS_ATP_RULESList>
```

Figure 1. Configuration Data XML

Single-Schema and Multischema Deployment Guidelines

The CDT allows you to compare and deploy data in single-schema and multischema environments. In both environments, data is organized into the following configuration groups:

- Metadata - required for database lookup of core configuration information. It is a set of unique tables across versions and performs “traffic direction” of connections across multischema, multi-version deployments.
- Configuration Data - consists of the following groups:
  - HUB Data - contains data and rules for the HUB.
  - Organization Driven Configuration Data - generally stores system or business rules such as sourcing rules, routing guide, and shipping preferences, as well as organizations.
- Master Data - consists of the following groups:
  - Organization Driven Master Data - contains data that is created through batch feeds and is often referenced by other organizations and users.
  - Ship Node Driven Master Data - contains master data relating to ship nodes.

For more information about configuration groups, refer to “CDT Configuration Groups and Driver Entities” on page 6.

Deploying Data in a Single-Schema Environment using the CDT

About this task

In a single-schema environment, data can be compared and deployed by organization.

To deploy data by organization, follow these steps:

Procedure

1. Deploy the Configuration Data Configuration Group for the DEFAULT organization and the DEFAULT organization's participating organizations.
2. Deploy the Configuration Data Configuration Group for organizations with configuration data that is referenced by other organizations, such as configuration inheritance, carriers, catalog/pricing organizations, etc.
3. Deploy the Master Data Configuration Group.

Results

For more information about deploying data, refer to “Starting the CDT in GUI Mode” on page 19.

CDT Single-Schema Deployment Examples

The following is the participant setup configured for examples 1 and 2:

- DEFAULT
- ORG1
- ORG2 - inherits configuration from ORG1
- ORG3 - defines its own configurations
- CARRIER1 - a carrier that participates with DEFAULT
- CARRIER2 - a carrier that participates with ORG2 and ORG3

Single-Schema Deployment Example 1:

About this task

To deploy ORG2 from source to target with Factory Setup:

Procedure

1. Deploy Configuration Data for DEFAULT organization and CARRIER1.
2. Deploy Configuration Data for ORG1, ORG2, and CARRIER2.

   CARRIER2 participates with ORG2. For this reason, CARRIER2 must be deployed with ORG2.
3. Deploy Master Data for ORG2.

Single-Schema Deployment Example 2:

About this task

To deploy ORG3 from source to target with Factory Setup:

Procedure

1. Deploy Configuration Data for DEFAULT and CARRIER1.
2. Deploy Configuration Data for ORG3 and CARRIER2.

3. Deploy Master Data for ORG3.

Deploying Data in a Multischema Environment using the CDT

About this task

In a multischema environment, data can be compared and deployed by organization or by colony. You must deploy the Metadata Configuration Group, Organization Data Configuration Group, and Master Data Configuration Group separately against the corresponding schemas. For more information about multischema configurations in general, see the *Selling and Fulfillment Foundation: Multitenant Enterprise Guide*.

To deploy data by organization or by colony, follow these steps:

Procedure

1. Deploy the Metadata Configuration Group. This step is necessary only when moving from a test environment to a production environment.
2. Deploy the Configuration Data Configuration Group for the DEFAULT organization and the DEFAULT organization's participating organizations.
3. Deploy the Configuration Data Configuration Group for organizations with configuration data that is referenced by other organizations, such as configuration inheritance, carriers, catalog/pricing organizations, etc.
4. Deploy the Configuration Data Group Configuration Group for other colonies or organizations belonging to other colonies.
5. Deploy the Master Data Configuration Group.
6. After deploying all data, run the Colony Map Synchronizer agent, as described in the "Time-Triggered Transaction Reference" appendix in the *Sterling Distributed Order Management: Configuration Guide*.

CDT Multischema Deployment Examples

The following is the participant setup configured for examples 1 and 2:

- DEFAULT COLONY
  - DEFAULT
  - ORG1
  - ORG2 - inherits configuration from ORG1
  - ORG3 - defines its own configurations
  - CARRIER1 - participates with DEFAULT
- COLONY1
  - ORG4
  - ORG5
  - CARRIER2 - participates with ORG4

Multischema Deployment Example 1:

About this task

To deploy ORG2 from source to target with Factory Setup:

Procedure

1. Deploy Configuration Data for DEFAULT organization and CARRIER1.
2. Deploy Configuration Data for ORG1 and ORG2.
3. Deploy Master Data for ORG2.

4. Run Colony May Synchronizer.

Multischema Deployment Example 2:
About this task

To deploy COLONY1 from source to target with Factory Setup:

Procedure

1. Deploy Configuration Data for DEFAULT organization and CARRIER1.
2. Deploy Configuration Data for COLONY1, which includes Configuration Data for ORG4, ORG5, and CARRIER2.

   CARRIER2 participates with ORG4. For this reason, CARRIER2 must be deployed with ORG4.
3. Deploy Master Data for COLONY1.
4. Run Colony Map Synchronizer.

CDT Configuration Groups and Driver Entities

The entire set of the Sterling Selling and Fulfillment Foundation configuration data is broken down into logical subsets called "configuration groups" and "driver entities". Configuration groups and driver entities are predefined and cannot be changed.

During the deployment process, if you need to perform more granular inserts, updates, and deletes so that your target database matches your source, you choose these configuration groups or driver entities.

Driver Entities

Most of the Sterling Selling and Fulfillment Foundation configuration data can be deployed starting with a logical entity, for example, an organization or a pipeline. These logical entities are called "driver entities". Driver entities represent the most granular level of information that can be deployed from the source to the target without loss of data integrity.

Only driver entities allow deployment at a record level. For other tables either of the following conditions apply:

- The table is completely deployed if it is not dependent on any driver entity.
- Only records corresponding to the driver entity are deployed.

Information about driver entities can be stored in multiple tables and when deploying an entity, data in all related tables is deployed together in one transaction boundary to preserve data consistency.

Configuration Groups

Logically related tables or driver entities are also grouped together into "configuration groups" that typically represent larger, significant logical data models within Sterling Selling and Fulfillment Foundation. Examples include the Business Process Model or the Participant Model. These groups are provided for convenience and for ease of navigation on the user interface.

Importing Externally Maintained Configuration Data

In your implementation of Sterling Selling and Fulfillment Foundation, you may be required to import certain data into your target that is not part of your source Sterling Selling and Fulfillment Foundation database. For these tables, you should not use CDT to deploy data as it does not have access to the correct data.

Best Practices

If you must use the Configuration Deployment Tool to deploy externally maintained data, the recommended way to handle this is to import this data into the source and then use CDT to deploy it into the target. This guarantees data integrity.

If you cannot import this data into your source database, Sterling Selling and Fulfillment Foundation supplies features that enable you to work with external data by ensuring that the target database either ignores these tables or appends them. Use the Ignore and Append-only features only if you have tried all other available options and only after subjecting your environment to rigorous testing.

CAUTION:
When using the Ignore or Append-only features, the CDT cannot guarantee the integrity of any external data. In order to ensure data integrity, CDT must have complete access to the configuration data.

Ignore

In cases where data in tables is maintained externally, you can omit these tables from the deployment operation by specifying a preference for them to be ignored.

Ignoring a table or a driver entity also automatically ignores all its dependent tables. However, there are some tables that store data for multiple driver entities and are present in multiple groups. An example of this is the YFS_GRAPH_UI table that contains data for pipelines, services and statuses. Ignoring one of these tables causes CDT to incorrectly mark the corresponding records for deletion.

Append-Only Mode

In cases where some tables are partially maintained externally, you can specify preferences to ensure that these tables are deployed in an "append-only" mode.

For append-only tables, the dependent tables are not ignored. Marking a table as append-only implies that only a few rows in the target database are maintained on the source system—other rows are externally imported. In such cases, it is extremely important that there is no overlap between the data present in the source and the external system. For example, if you maintain your shipping nodes in the source database and import store information directly into the target, you must not have any stores in the source database. This leads to unpredictable results.

How to Deploy Custom Tables Using the CDT

The CDT automatically deploys configuration tables and extensions defined within the Sterling Selling and Fulfillment Foundation database framework. If you have custom (non-Sterling Selling and Fulfillment Foundation) configuration tables defined in your installation, CDT needs to be specially configured to deploy these tables. To enable CDT to deploy these tables, the tables need to be registered with CDT by creating a special custom deployment XML file, called cdt_custom.xml. A sample of this file can be found in your <INSTALL_DIR>/resources/ydkresources directory. This file defines a group named "Custom Tables" and should include a list of your custom tables. CDT automatically compares, displays, and deploys changes to custom records for all tables that have one or more primary key columns.

This tool does not support custom tables as drivers or the representation of custom tables in a dependency tree structure. As a result, all custom tables can only be deployed together as part of the "Custom Tables" group. It also does not support custom tables without a primary key.

The cdt_custom.xml file contains the following:

```
<Group Name="Custom Tables">
   <Table Name="CUSTOM_CONFIG_TABLE_1"/>
   <Table Name="CUSTOM_CONFIG_TABLE_2"/>
</Group>
```

How the CDT Handles Foreign Key Checks

The CDT enforces data consistency by deploying all related tables that define an entity together in one operation. In addition, to ensure data integrity, the CDT also checks the required foreign key constraints for each table - which could potentially be defined for a table in a completely different group. Therefore, when deploying a small subset of data, it is possible that you may see error messages indicating foreign key constraint violations if the corresponding data in the independent table is not being deployed in the same operation. In this case, you should try deploying a bigger set of data. Note that foreign key constraints are not defined or checked for custom tables.

To provide the best performance, foreign key constraints are not checked when deploying the complete Sterling Selling and Fulfillment Foundation configuration.

Transform Data Using the CDT

Frequently, development and production environments have different values for network settings such as server names and IP addresses. Some configuration data tables in the Sterling Selling and Fulfillment Foundation store host names, IP addresses, and URLs. While these are valid for your source environment, when deploying this data into the target environment, the configuration must be updated with the corresponding values applicable to the target environment. The CDT enables you to automatically transform these data elements into target-appropriate values by letting you specify transformations to be carried out on the source data *before* it is deployed into the target.

CDT Resource Files

The CDT-specific resource files are located in the <INSTALL_DIR>/resources/ydkresources directory. These resource files allow you to modify the CDT preference settings, database-specific settings, and so forth.

cdt_custom.xml.sample

The cdt_custom.xml.sample is a sample of the special custom deployment XML file that can be used to deploy the custom tables defined in your installation. For more information about deploying custom tables, see “How to Deploy Custom Tables Using the CDT” on page 7.

cdt_dbdefaults.properties

This cdt_dbdefaults.properties file is used to provide extra database pool options for given database types. It contains default values for various databases that are used while configuring the CDT. The CDT uses this file to handle the database connections.

ydkprefs.xml

The `ydkprefs.xml` file contains the preference settings defined for the CDT and is created when you run the GUI-based CDT. It contains the configure preferences (such as a Reports Directory) and parameters that determine the behavior of the comparison operation of the CDT.

ydkprefs.xml.sample

The `ydkprefs.xml.sample` file is automatically included in the Sterling Selling and Fulfillment Foundation installation. You can edit it to specify configuration preferences (such as a Reports Directory) and parameters that determine the comparison operation behavior of the CDT. When finished modifying this sample file, rename it to `ydkprefs.xml`.

Chapter 3. Understanding the CDT Interface

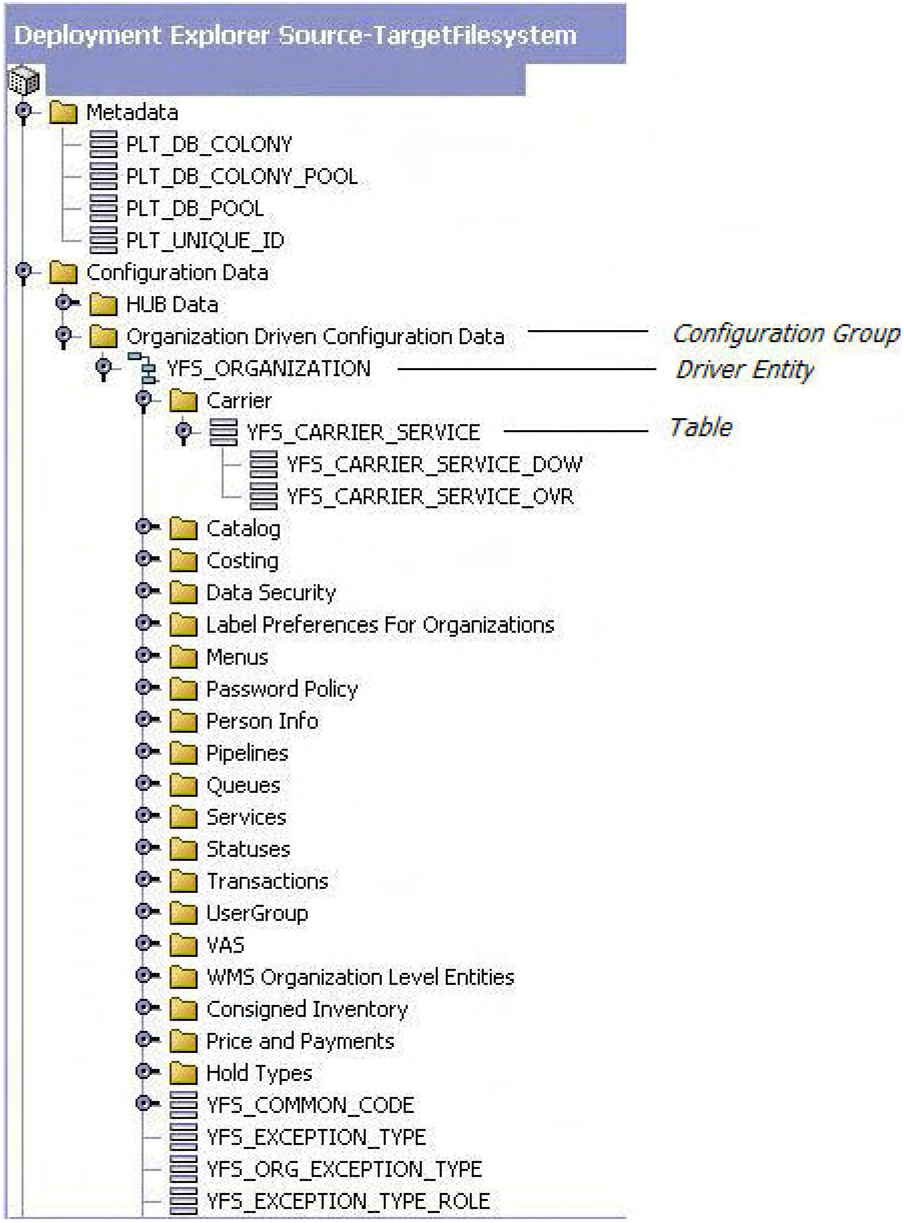
Understanding the CDT Interface

When the Configuration Deployment Tool starts, it prompts you to specify the details about your source and target databases to use during the session. After you have successfully connected to your source and target databases, the Deployment Explorer window appears.

CDT Deployment Explorer

The Deployment Explorer window displays the list of configuration groups, driver entities, and tables that can be deployed. The names you define for the source and target databases are displayed in the heading panel.

Each time you log into the Configuration Deployment Tool there is one instance of this window.



Configuration Group

Driver Entity

Table

Figure 2. Deployment Explorer

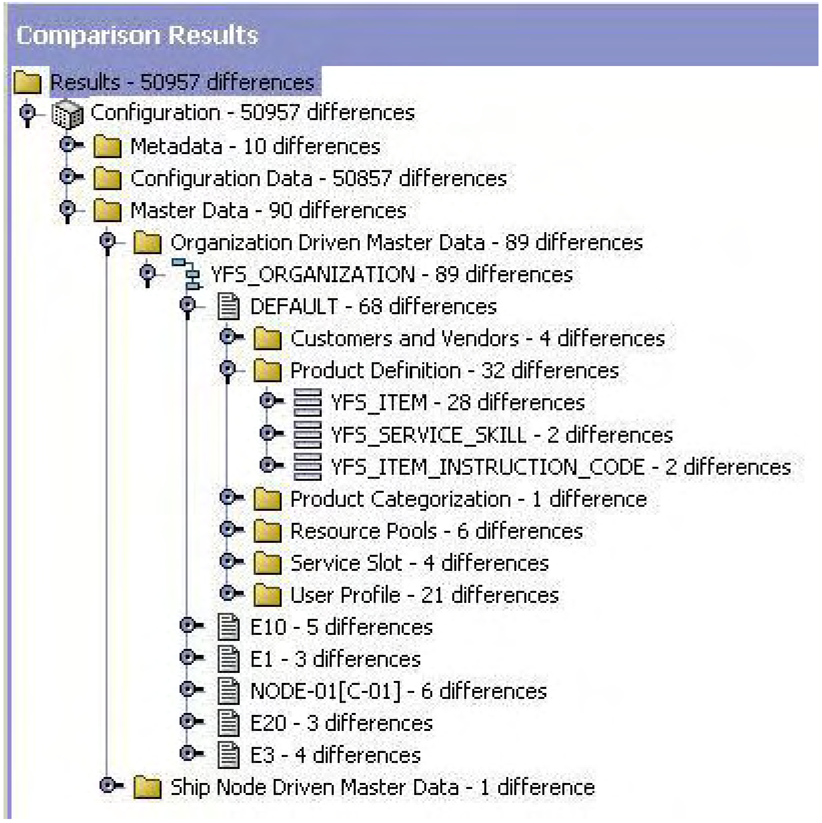
You can choose the configuration group or the driver entity that you want to compare between the source and target databases.

During the compare operation, the progress and the results of the comparison operation are displayed in the Comparison Results window and in the Status panel.

CDT Comparison Results Window

The Comparison Results window displays the outcome of the comparison between the source and target databases.

The Comparison Results window displays information pertaining to the current session. Only one Comparison Results window can be displayed during each session. After viewing the results of one comparison, you must close the window before you can compare a different set of tables.



```
Comparison Results
Results - 50957 differences
  Configuration - 50957 differences
    Metadata - 10 differences
    Configuration Data - 50857 differences
    Master Data - 90 differences
      Organization Driven Master Data - 89 differences
        YFS_ORGANIZATION - 89 differences
          DEFAULT - 68 differences
            Customers and Vendors - 4 differences
            Product Definition - 32 differences
              YFS_ITEM - 28 differences
              YFS_SERVICE_SKILL - 2 differences
              YFS_ITEM_INSTRUCTION_CODE - 2 differences
            Product Categorization - 1 difference
            Resource Pools - 6 differences
            Service Slot - 4 differences
            User Profile - 21 differences
          E10 - 5 differences
          E1 - 3 differences
          NODE-01[C-01] - 6 differences
          E20 - 3 differences
          E3 - 4 differences
      Ship Node Driven Master Data - 1 difference
```

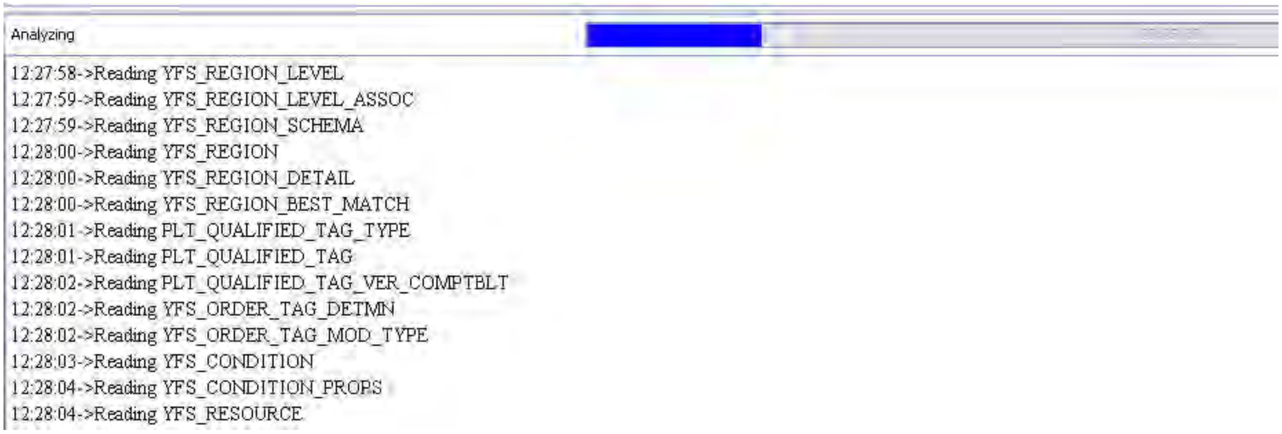
Figure 3. Comparison Results Window

After generating comparison results, you can carry out any one of the following tasks:

- Generate a report of the differences
- View the details of each difference
- Deploy configuration data from the source database into the target database

Deployment Tool Status Panel

The Status panel displays information about operations while they are carried out.



```
Analyzing
12:27:58->Reading YFS_REGION_LEVEL
12:27:59->Reading YFS_REGION_LEVEL_ASSOC
12:27:59->Reading YFS_REGION_SCHEMA
12:28:00->Reading YFS_REGION
12:28:00->Reading YFS_REGION_DETAIL
12:28:00->Reading YFS_REGION_BEST_MATCH
12:28:01->Reading PLT_QUALIFIED_TAG_TYPE
12:28:01->Reading PLT_QUALIFIED_TAG
12:28:02->Reading PLT_QUALIFIED_TAG_VER_COMPTBLT
12:28:02->Reading YFS_ORDER_TAG_DETMN
12:28:02->Reading YFS_ORDER_TAG_MOD_TYPE
12:28:03->Reading YFS_CONDITION
12:28:04->Reading YFS_CONDITION_PROPS
12:28:04->Reading YFS_RESOURCE
```

Figure 4. Configuration Deployment Tool Status Panel

Chapter 4. Before Using the Configuration Deployment Tool

CDT System Requirements

The RAM requirements of the Configuration Deployment Tool depend on the size of your database and the distribution of your configuration data.

Time Estimates

The time required for the Configuration Deployment Tool to perform comparison and deployment tasks varies according to your system resources and the size and distribution of your configuration data. For example, processing time may increase when there are many records in a table that are referenced by foreign key constraints from other tables, or when there are many records in a table that serves as a driver entity.

It is recommended that you migrate no more than 200,000 records per deployment.

During tests of the Configuration Deployment Tool, Sterling Selling and Fulfillment Foundation measures the amount of time it takes to perform tasks on a Pentium class machine with 512 MB of RAM and running at 550 MHz. The Configuration Deployment Tool performed as described in Table 1.

Table 1. Time Estimates using the Configuration Deployment Tool

Task	Description of Databases	Time
Comparison	Source database - 110,000 records Target database - 110,000 records	7 minutes
Comparison	Source database - 110,000 records Target database - empty	4 minutes
Deployment	110,000 differences	11 minutes

Note: Tests of the Configuration Deployment Tool on a 64-bit machine with 3 GB of RAM and running at 2 GHz demonstrate that the CDT can compare and deploy a maximum of 750,000 records per enrollment.

Installation

The Configuration Deployment Tool is installed automatically during the installation of Sterling Selling and Fulfillment Foundation.

Environment State

The Configuration Deployment Tool assumes that your source and target environments match exactly in the following respects:

- Release of Sterling Selling and Fulfillment Foundation (including fix packs)
- Release of JDBC drivers
- Release of database software
- Database structure (schema objects such as tables, indexes, and sequences)

Note: Before using the Configuration Deployment Tool, make sure that you have run the Factory Setup on both source and target environments.

Because the Configuration Deployment Tool is used by technical professionals for tasks they perform on an occasional basis, it is **not** localizable or customizable. However, you can specify configuration preferences.

CDT Security Strategy

The Configuration Deployment Tool makes use of the user authentication and authorization supplied by your database provider. Access control and authorization are not specified through the Configuration Deployment Tool.

Ensure that the person using the Configuration Deployment Tool has sufficient authentication privileges (select, insert, update, and delete) for both databases; full DBA privileges are not required.

CDT Change Management Strategy

The Configuration Deployment Tool does **not** enforce checks to restrict configuration data modification on the source or the target schemas using other means. You must develop and enforce your own methodology.

For example, if you use the Configuration Deployment Tool to migrate data from staging to production, it is **not** expected that the configuration in production is modified by means other than this tool. In such a case those changes are overwritten the next time CDT is run. Also, this could potentially lead to data integrity issues if the changes are performed in either the source or target while CDT is being run.

The Configuration Deployment Tool is **not** supported for implementations where configuration data is directly modified in production using the Applications Manager or any other means. For exceptional cases like urgent or critical fixes to configuration data in production, you must update the staging database with the same changes.

CDT Rollback Strategy

To prevent application failure and downtime, implement a rigorous rollback methodology that involves creating a backup snapshot of your configuration data in production before you use the Configuration Deployment Tool to deploy changes. This backup is accomplished by using the database-specific export and import utilities. Sterling Selling and Fulfillment Foundation provides samples for Oracle, DB2®, and Microsoft SQL Server databases that you can customize for your own use. For more information on data rollback, see “Data Rollback Scripts” on page 59

About the CDT and Upgrades and Maintenance

Using the Configuration Deployment Tool should not impact the methodology for applying upgrades or fix packs in a multi-step staging environment.

The upgrade methodology being followed should not change for environments already set up for staging before production. However, the Configuration Deployment Tool by itself does **not** provide support for all of the processes and methodologies required for supporting a multi-step application staging and deployment environment because it is only capable of deploying configuration data.

The process of applying product upgrades and fix packs is especially complex in an environment where the staging area must be kept synchronized with production. One way to keep these environments harmonious is to apply software fix packs to both systems simultaneously and reverse deploy the data upgrades. This is because application data upgrades may behave differently and produce different results based on the transactional data they encounter. If this application data upgrade is run independently on production and staging, the results may be significantly different as a result of the differences in transactional data that the upgrade program encountered. In such a case, the production snapshot should be treated as the baseline and reverse deployed into staging. This can be accomplished by configuring your production database as the source and your staging database as the target.

Example Upgrade Scenario

In an example upgrade scenario, Sterling Selling and Fulfillment Foundation introduces a feature that recognizes the various attributes for order types. For example, an Order_type "URGENT" implies that the order should be displayed in the user interface with a specific icon that enables you to distinguish it from other orders. However, in previous releases, you may have been using the **Order_type** field to classify orders into other types because this field was designed for order classification.

When Sterling Selling and Fulfillment Foundation provides an upgrade toolkit, one component of the toolkit handles upgrades to the **Order_type** field.

The upgrade logic may flow as follows:

1. Read all the distinct values of the **Order_type** field from the `YFS_ORDER` table.
2. For each different Order_type in the your system, create entries in the Order_Type_Master configuration table and assign a default icon to it.

If this data upgrade is run on the staging system, it will **not** find any orders, so the Order_Type_Master table will only contain "URGENT" which is provided by default.

However, when the same data upgrade is run in production, the Order_Type_Master table will contain multiple entries, one for each type of order that is in the transaction database.

Then, when the Configuration Deployment Tool is run again, it marks all of these new records for deletion because the source is assumed to be the most up-to-date configuration.

As a result, you should design upgrade kits or service packs for transaction dependent configuration data as follows:

1. The upgrade kit (or service pack) should have one script to prepare input for upgrade of transaction-dependent configuration data (for example, prepared list of distinct order types). Then you can run this script on the production database. You can also run this script in the test database and can take the union of the two.

2. The next step in the upgrade should use this as input and upgrade the configuration data accordingly. For example, inserting into ORDER_TYPE_MASTER table.

If you have identified any changes in the configuration data, please contact IBM® Technical Support.

Chapter 5. Setting Up and Using the CDT in GUI Mode

Setting Up and Using the CDT in GUI Mode

You can configure preference settings for the Configuration Deployment Tool and run it from a GUI.

CDT Command-Line Arguments in GUI Mode

When you run the Configuration Deployment Tool, you can optionally use any of the command parameters shown in Table 2:

Table 2. Java ydk Command-Line Arguments

Arguments	Description
IgnoreMissingTables	Specifies that when you compare source and target databases, you want to ignore tables that might be missing in the target schema. When comparing databases in a multischema deployment, ensure that you specify the `IgnoreMissingTables` parameter. For example, when comparing a Test Configuration schema with a Production Configuration schema, some Master tables, such as YFS_CUSTOMER and YFS_USER, do not exist in either the source or target schema. In this case, the CDT throws "table not found" errors. However, you can enable the CDT to ignore missing tables by passing `-IgnoreMissingTables` as `Y`. Example: Windows: `ydk.cmd -IgnoreMissingTables Y` Linux/UNIX: `ydk.sh -IgnoreMissingTables Y`
ConfigDbDir	Indicates the directory that contains `config-db.xml` and other related files. By default, the ConfigDbDir argument looks for the `config-db.xml` file in the `<INSTALL_DIR>/database/cdt` directory. Example: Windows: `ydk.cmd -ConfigDbDir <directory_path>` Linux/UNIX: `ydk.sh -ConfigDbDir <directory_path>`

Starting the CDT in GUI Mode

About this task

Consider the following before you begin:

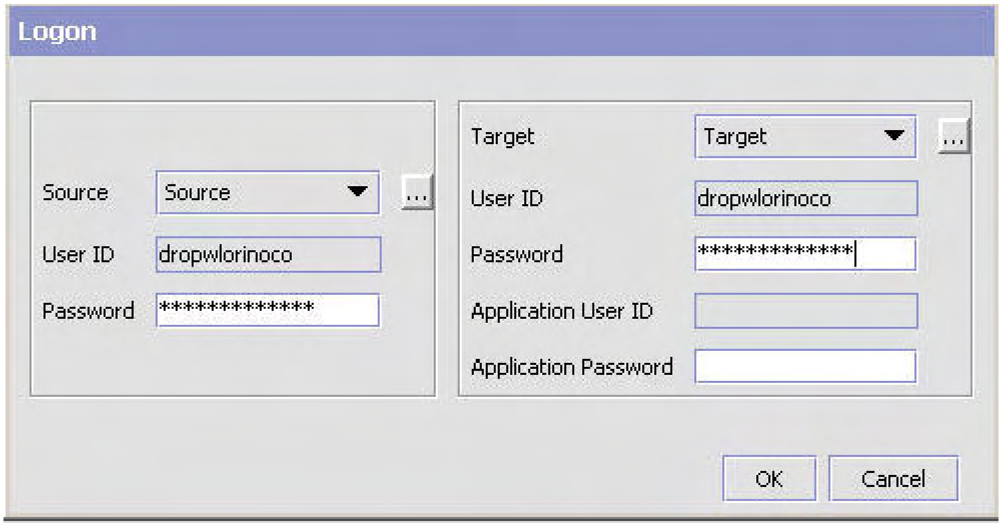
- Most preference settings are tied to a specific source and target database combination. If you change the name of either the source or target database, any existing settings are not used. To return to your old settings, revert to the old source and target database name.
- To run the UI for the Configuration Deployment Tool on a UNIX server from a windows client, the UNIX server must have XWindows installed on it.

- The Windows console displays WorkBench startup information. Do not close the console while the WorkBench is running. Closing the console closes the tool, and your work is lost.

To start the Configuration Deployment Tool:

Procedure

1. Start the Sterling Selling and Fulfillment Foundation Development and Deployment WorkBench. On Microsoft Windows, run the `ydk.cmd` script for Windows (`ydk.sh` for UNIX/Linux) from the `<INSTALL_DIR>\bin` directory.
2. From the Development and Deployment WorkBench menu, choose **Tools** → **Deployment** → **Configuration Data Deployment**. This opens the Configuration Deployment Tool Logon dialog box.



Logon

Source: Source

User ID: dropwlorinoco

Password: **************

Target: Target

User ID: dropwlorinoco

Password: **************

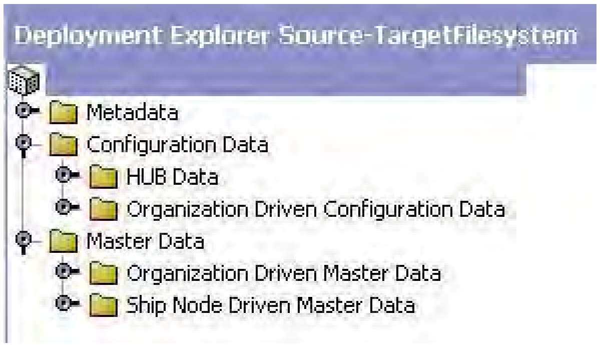
Application User ID:

Application Password:

Figure 5. Configuration Deployment Tool Logon Dialog Box

3. Choose the **Source** button and enter the values appropriate for the source database. Then choose the **Target** button and enter the values appropriate for the target database.

   In a multischema environment, you must migrate each configuration group, one at a time. if you are migrating HUB Data and Organization Driven Configuration Data, these two configuration groups must be your sources. These groups are displayed in the Deployment Explorer and are provided as part of Sterling Selling and Fulfillment FoundationSterling Application Platform.


Deployment Explorer Source-TargetFilesystem

- Metadata
- Configuration Data
  - HUB Data
  - Organization Driven Configuration Data
- Master Data
  - Organization Driven Master Data
  - Ship Node Driven Master Data

Figure 6. Deployment Explorer Source-Target File System

Your configuration data is in the HUB Data and Organization Driven Configuration Data groups, so the target for each of these should be the Configuration Schema. If you are migrating Organization Driven Master Data, the target should be the Master Data Schema. To migrate Metadata, the target should be the Metadata schema.

When you are finished, close the dialog box. The values you specified are saved automatically and persist from one session to the next.

In the Source database and Target database windows, specify the applicable values as described in Table 3.

Table 3. Configuration Deployment Tool Logon Dialog Box

Field	Detail
Name	Specify a logical database identifier. For the source, specify the database you want to copy data from. For the target, specify the database to write the data to.
className	Specify the class name of your database driver as follows: • If using Oracle, set to: `oracle.jdbc.OracleDriver` • If using Microsoft SQL Server 2005/2008, set to: `com.microsoft.sqlserver.jdbc.SQLServerDriver` • If using DB2, set to: `com.ibm.db2.jcc.DB2Driver`
jdbcURL	Specify the URL to connect to the database: • If using Oracle, set to: `jdbc:oracle:thin:@<DatabaseServerHostname/IPaddress>:<TNSListenerPortNumber>:<DatabaseSID>.` • If using Microsoft SQL Server 2005/2008, set to: `jdbc:sqlserver://<Database Server Hostname>:<Port Number>;DatabaseName=<Database name>.` • If using DB2, set to: `jdbc:db2://<Database Server Hostname>:<Port Number>/<Database name>.`
dbType	Specify the type of database you are running. Enter it in all lower case, as shown: • For Oracle, specify `oracle` • For Microsoft SQL Server, specify `sqlserver` • For DB2, specify `db2` • For an XML datasource, specify `xml`
folder	If using an XML datasource, specify the complete path of the folder location for the XML files.

Table 3. Configuration Deployment Tool Logon Dialog Box (continued)

Field	Detail
httpurl	Only applicable for the target database. Specify a URL for the application server whose data cache is to be refreshed after data is deployed into the target database. Use the syntax: `http://<hostname/ip-address>:<port-number>/<Application>/interop/InteropHttpServlet`, where `<hostname/IP-address>` is the server address where the application is running, `<port-number>` is the port on which the application is running, and `<Application>` is the name of the application; for example, `smcfs`.
httpuser	Specify the surname associated with the application server whose data cache is to be refreshed after data is deployed into the target database.
schema	Specify the schema owner as follows: • If you are using Oracle or DB2 database, and the user you specify is different from the Sterling Selling and Fulfillment Foundation schema owner, specify the owner of the Sterling Selling and Fulfillment Foundation schema. • If you are using Microsoft SQL Server, leave this blank.
user	Specify the user name associated with the database.

4. In the Logon dialog box, enter the passwords associated with the user names. The Deployment Explorer window displays.

Stopping the CDT in GUI Mode

About this task

To stop the Configuration Deployment Tool:

Procedure

From the Sterling Selling and Fulfillment Foundation Development and Deployment WorkBench menu, choose **File** → **Exit** .
This closes the Configuration Deployment Tool and the Windows console.

Setting CDT Preference Settings in GUI Mode

About this task

You can configure preferences (such as a reports directory) and parameters that determine the behavior of the comparison operation. When you modify these properties, the changes persist, so you do not need to reset them each time you use CDT. These changes are saved in the `<INSTALL_DIR>/resources/ydkresources/ydkprefs.xml` file.

The `ydkprefs.xml` is created the first time you start the CDT from the GUI. Alternatively, you can manually create the `ydkprefs.xml` file without starting the CDT from the GUI, which allows you to invoke the `ydkprefs.xml` file from the command line. For information about manually creating and editing the `ydkprefs.xml` file, see “Setting CDT Preference Settings in Text Mode” on page 27.

To specify Configuration Deployment Tool settings:

Procedure

1. Start the Configuration Deployment Tool. For more information about starting the Configuration Deployment Tool, see “Starting the CDT in GUI Mode” on page 19.
2. From the **Deployment Explorer** action bar, choose the **Preferences** icon.
3. In the Preferences window, fill in values using the descriptions provided in Table 4.

Table 4. Configuration Deployment Tool Preference Settings

Control	**Description**
Settings Tab	
Reports Directory (for CDT)	If you are running CDT, specify the absolute path where you want reports to be generated.
Reports Directory (for CDV Tool)	If you are running CDV Tool, specify the absolute path where you want to export the comparison results report.
Custom Deployment Class	Specify the name of the class that should be invoked for deploying custom tables not handled by CDT.
Max Changes to Display	Specify the maximum number of differences to be displayed. The default display number is 100.
Audit Version Deployment	
Validate Old Values	If you select this check box, the system detects conflict based on the expected old value of an attribute of a record of the obtained changes from the source database against the current value of the corresponding record in the target database.
Validate Lockid	If you select this check box, the system detects conflict based on the expected Lockid of a record of the obtained changes from the source database against the current Lockid of the corresponding record in the target database.
Validate Record Exists Before Delete	If you select this check box, the system validates the record exists in the target database before it attempts to delete it. If the record is not there, the operation will be marked as a conflict.
Transformations Tab	
Table Element	Tables that can be added or deleted.
Table Name Attribute	Specify the name of the table on which you want to carry out the transformation. The syntax and case must match the name of the table used in the Sterling Selling and Fulfillment Foundation ERDs. Custom tables cannot be transformed. Choose the **Details** icon to specify a value.
Column Element	Columns that can be added or deleted.
Column Name Attribute	Specify the name of the column containing the data to be transformed. The syntax and case must match the name of the column used in the Sterling Selling and Fulfillment Foundation ERDs. Extended columns can be transformed. Choose the Details icon to specify a value.
Transform Element	Define the transformation for this column. For each column, you can define one or more transformations. These transformations are applied to data in this column in sequential order. You can specify multiple transformations for each column, using the delete action to remove the parent element.

Table 4. Configuration Deployment Tool Preference Settings (continued)

Control	Description
Match Attribute	Specify the pattern to search for in the source data. All matching occurrences of this pattern are replaced with the value specified in the Replace attribute. Choose the Details icon to specify a value.
Replace Attribute	Specify the value to replace the pattern with. Choose the Details icon to specify a value.
XPath Attribute	Conditional. If the column to be transformed contains non-XML data, you do not need to specify this XPath attribute. However, some configuration information in Sterling Selling and Fulfillment Foundation is stored as XML in the database. If the column to be transformed contains XML data, use this attribute to specify the location of the exact attribute to be transformed. Use the syntax: `xml:/Configuration/Connection/Host/@IPAddress`. Choose the **Details** icon to specify a value.
Append-Only Tables Tab	
Append-only Tables	Specify configuration table, if any, in which *some* rows maintain data that is external to Sterling Selling and Fulfillment Foundation. This prevents the data from being deleted during deployment. Specify that table and all of its dependent tables. **Note:** Rows that are maintained externally should never be present in your source database, since this can lead to unpredictable results.
Ignore Tables Tab	
Ignore Tables	Specify any external configuration tables that you do not want the tool to deploy from the source to the target. Ignoring a table automatically ignores all dependent tables as well.

Chapter 6. Setting Up and Using the CDT in Text Mode

Setting Up and Using the CDT in Text Mode

You can run and configure preference settings for the Configuration Deployment Tool from a command line.

CDT Command-Line Arguments in Text Mode

When you run the Configuration Deployment Tool, you can optionally use any of the command parameters shown in Table 5:

Table 5. Java cdtshell Script Command-Line Arguments

Arguments	Description
ColonyId	Specifies the Colony ID that you want to compare or deploy. You can pass only one colony at a time. Example: Windows: `cdtshell.cmd -ColonyId E1` Linux/UNIX: `cdtshell.sh -ColonyId E1`
IgnoreMissingTables	Specifies that when you compare source and target databases, you want to ignore tables that might be missing in the target schema. When comparing databases in a multischema deployment, ensure that you specify the `IgnoreMissingTables` argument. For example, when comparing a Test Configuration schema with a Production Configuration schema, some Master tables, such as YFS_CUSTOMER and YFS_USER, do not exist in either the source or target schema. In this case, the CDT throws "table not found" errors. However, you can enable the CDT to ignore missing tables by passing `-IgnoreMissingTables` as `Y`. Example: Windows: `cdtshell.cmd -IgnoreMissingTables Y` Linux/UNIX: `cdtshell.sh -IgnoreMissingTables Y`

LabelId	Specifies the Label Id value that is used to create labels such as BEGIN_<LabelId> and END_<LabelId>, before and after the deployment, respectively. If this argument is not passed, labels are not created.
	Example:
	`Windows: cdtshell.cmd -LabelId OrgA1`
	`Linux/UNIX: cdtshell.sh -LabelId OrgA1`

Table 5. Java cdtshell Script Command-Line Arguments (continued)

Arguments	Description
CompareOrganizationCode	Specifies the organizations you want to compare based on the organization codes passed, as defined in the `config-db.xml`. If you do not pass the `CompareOrganizationCode` argument, the corresponding filter is not used. Use a comma-delineated format to specify organizations to compare, such as Org-1, Org-2. If you use the `CompareOrganizationCode` argument and specify no organizations, the CDT compares all organizations. It is recommended that all participating organizations are compared together. For example, if two organizations, Org-01 and Org-02, participate with two ship nodes, Node-01 and Node-02, you should specify Org-1,Org-2,Node-01,Node-02. Example: Windows: `ydk.cmd -CompareOrganizationCode Org-1,Org-2,Node-01,Node-02` Linux/UNIX: `ydk.sh -CompareOrganizationCode Org-01,Org-02,Node-01,Node-02`

Running the CDT in Text Mode

About this task

This section explains how to run the Configuration Deployment Tool from a command line, using a text-based (non-GUI) interface.

To run the Configuration Deployment Tool from command line:

Procedure

1. Edit the `<INSTALL_DIR>/bin/cdtshell.cmd.in` script for windows (`cdtshell.sh.in` for UNIX/Linux) and set the following properties:

   Property

   Description

   SOURCE_DB

   The logical identifier for the source database. This value corresponds to the `Name` setting in the `<SourceDatabases>` element in the ydkprefs.xml file.

   TARGET_DB

   The logical identifier for the target database. This value corresponds to the `Name` setting in the `<TargetDatabases>` element in the ydkprefs.xml file.

   SOURCE_PASSWORD

   Specify the password required for the source database.

   Note: This parameter is not required to be set if the source is an XML folder.

   TARGET_PASSWORD

   Specify the password required for the target database.

Note: This parameter is not required to be set if the source is an XML folder.

TARGET_HTTP_PASSWORD
Specify the password of the http user required to refresh the cache on the server.

Note: This parameter is not required if there is no cache to refresh on the server.

2. Run the `<INSTALL_DIR>/bin/setupfiles.cmd` script for windows (`setupfiles.sh` for UNIX/Linux).
3. Configure CDT preferences and parameters for the comparison operation. The `ydkprefs.xml` file determines CDT preferences and is automatically created when you run the CDT from the GUI; however, you can also manually create and edit the `ydkprefs.xml` file without using the CDT GUI. For information about manually creating and editing the `ydkprefs.xml` file, see "Setting CDT Preference Settings in Text Mode."
4. Run the `cdtshell.cmd` script for windows (`cdtshell.sh` for UNIX/Linux) from the `<INSTALL_DIR>\bin` directory.

Setting CDT Preference Settings in Text Mode

About this task

You can manually configure preferences and parameter settings that determine the behavior of the comparison operation by editing the `ydkprefs.xml.sample` file and saving the file as `ydkprefs.xml.`

To configure Configuration Deployment Tool preference settings:

Procedure

1. Rename the `<INSTALL_DIR>/resources/ydkresources/ydkprefs.xml.sample` file to `ydkprefs.xml`.
2. In `ydkprefs.xml`, specify parameters for CDT preference elements. Table 6 describes the elements in `ydkprefs.xml.`

Table 6. Configuration Deployment Tool Preference Settings

Element	**Description**
<Settings> Element	
CustomEntityClass	Specify the name of the class that should be invoked for deploying custom tables not handled by CDT. **Note:** If you don't want to use this field, set the value of this field to null. For example, CustomEntityClass="".
MaxChangesToDisplay	Specify the maximum number of differences to be displayed. The default display number is 100. **Note:** If you don't want to use this field, set the value of this field to null. For example, MaxChangesToDisplay="".
ReportsDir	Specify the absolute path where you want reports to be generated. **Note:** If you don't want to use this field, set the value of this field to null. For example, ReportsDir="".

Table 6. Configuration Deployment Tool Preference Settings (continued)

Element	**Description**
ValidateLockid	Specify `Y`, if you want the system to detect conflict based on the expected Lockid of a record of the obtained changes from the source database against the current Lockid of the corresponding record in the target database. By default, it is set to N.
ValidateOldValues	Specify `Y`, if you want the system to detect conflict based on the expected old value of an attribute of a record of the obtained changes from the source database against the current value of the corresponding record in the target database. By default, it is set to Y.
ValidateRecordExistsBeforeDelete	Specify `Y`, if want the system to validate the record exists in the target database before it attempts to delete it. If the record is not there, the operation will be marked as a conflict. By default, it is set to Y.
<SourceDatabases> Element	
Name	Specify a logical database identifier. For the source, specify the database you want to copy data from. For the target, specify the database to write the data to.
className	Specify the class name of your database driver as follows: • If using Oracle, set to: `oracle.jdbc.OracleDriver` • If using Microsoft SQL Server 2005/2008, set to: `com.microsoft.sqlserver.jdbc.SQLServerDriver` • If using DB2, set to: `com.ibm.db2.jcc.DB2Driver`
dbType	Specify the type of database you are running. Enter it in all lower case, as shown: • For Oracle, specify `oracle` • For Microsoft SQL Server, specify `sqlserver` • For DB2, specify `db2` • For an XML datasource, specify `xml`
folder	If using an XML datasource, specify the complete path of the folder location for the XML files. **Note:** If the source database is a database schema, you must set the value of this attribute to null. For example, folder="".
jdbcURL	Specify the URL to connect to the database: • If using Oracle, set to: `jdbc:oracle:thin:@<DatabaseServerHostname/IPaddress>:<TNSListenerPortNumber>:<DatabaseSID>.` • If using Microsoft SQL Server 2005/2008, set to: `jdbc:sqlserver://<Database Server Hostname>:<Port Number>;DatabaseName=<Database name>.` • If using DB2, set to: `jdbc:db2://<Database Server Hostname>:<Port Number>/<Database name>.`

Table 6. Configuration Deployment Tool Preference Settings (continued)

Element	Description
schema	Specify the schema owner as follows: • If you are using Oracle or DB2 database, and the user you specify is different from the Sterling Selling and Fulfillment Foundation schema owner, specify the owner of the Sterling Selling and Fulfillment Foundation schema. • If you are using Microsoft SQL Server, leave this field blank. For example, schema="". **Note:** For Oracle and DB2 databases, leave this field blank if the schema name is same as the username.
user	Specify the user name associated with the database.
<TargetDatabases> Element	
For Name, className, dbType, folder, jdbcURL, schema, and user attributes descriptions, see the <SourceDatabases> Element section in this table.	
httpurl	Only applicable for the target database. Specify a URL for the application server whose data cache is to be refreshed after data is deployed into the target database. Use the syntax: `http://<hostname/ip-address>:<port-number>/<Application>/interop/InteropHttpServlet`, where `<hostname/IP-address>` is the server address where the application is running, `<port-number>` is the port on which the application is running, and `<Application>` is the name of the application; for example, `smcfs`.
httpuser	Specify the username associated with the application server whose data cache is to be refreshed after data is deployed into the target database.
<SourceTargetPrefs> Element	
SourceDatabase	Specify the name of the source database.
TargetDatabase	Specify the name of the target database.
<Transformations> Element Tables and Columns that can be added or deleted.	
<Table> Element's Name Attribute	Specify the name of the table on which you want to carry out the transformation. The syntax and case must match the name of the table used in the Sterling Selling and Fulfillment Foundation ERDs. Custom tables cannot be transformed. Choose the Details icon to specify a value. **Note:** If you don't want to use this field, set the value of this field to null. For example, Name="".
<Column> Element's Name Attribute	Specify the name of the column containing the data to be transformed. The syntax and case must match the name of the column used in the Sterling Selling and Fulfillment Foundation ERDs. Extended columns can be transformed. Choose the Details icon to specify a value. **Note:** If you don't want to use this field, set the value of this field to null. For example, Name="".
<Transform> Element Define the transformation for this column. For each column, you can define one or more transformations. These transformations are applied to data in this column in sequential order. You can specify multiple transformations for each column, using the delete action to remove the parent element.	
Match	Specify the pattern to search for in the source data. All matching occurrences of this pattern are replaced with the value specified in the Replace attribute. Choose the Details icon to specify a value.
Replace	Specify the value to replace the pattern with. Choose the Details icon to specify a value.

Table 6. Configuration Deployment Tool Preference Settings (continued)

Element	Description
XPath	Conditional. If the column to be transformed contains non-XML data, you do not need to specify this XPath attribute. However, some configuration information in Sterling Selling and Fulfillment Foundation is stored as XML in the database. If the column to be transformed contains XML data, use this attribute to specify the location of the exact attribute to be transformed. Use the syntax: `xml:/Configuration/Connection/Host/@IPAddress`. Choose the Details icon to specify a value.
<Ignore> Element	
<Table> Element's Name Attribute	Specify any external configuration tables that you do not want the tool to deploy from the source to the target. Ignoring a table automatically ignores all dependent tables as well. **Note:** If you don't want to use this field, set the value of this field to null. For example, Name="".
<AppendOnly> Element	
<Table> Element's Name Attribute	Specify configuration table, if any, in which *some* rows maintain data that is external to Sterling Selling and Fulfillment Foundation. This prevents the data from being deleted during deployment. Specify that table and all of its dependent tables. **Note:** Rows that are maintained externally should never be present in your source database, since this can lead to unpredictable results. **Note:** If you don't want to use this field, set the value of this field to null. For example, Namer="".

Chapter 7. About Using the CDT to Transform Elements

About Using the CDT to Transform Elements

When deploying data from one database instance to another, you can override the values of certain data elements. For example, if your source and target environment network settings (host names, port numbers, and IP addresses) are different, the Configuration Deployment Tool can transform the settings in order to make them appropriate for the target environment.

Transformations are carried out as a pattern match and replace the data in the source database before it is deployed into the target.

The match and replace are carried out for the complete string literal and no wild card search for characters is allowed.

For example, consider the following configuration XML in the source database:

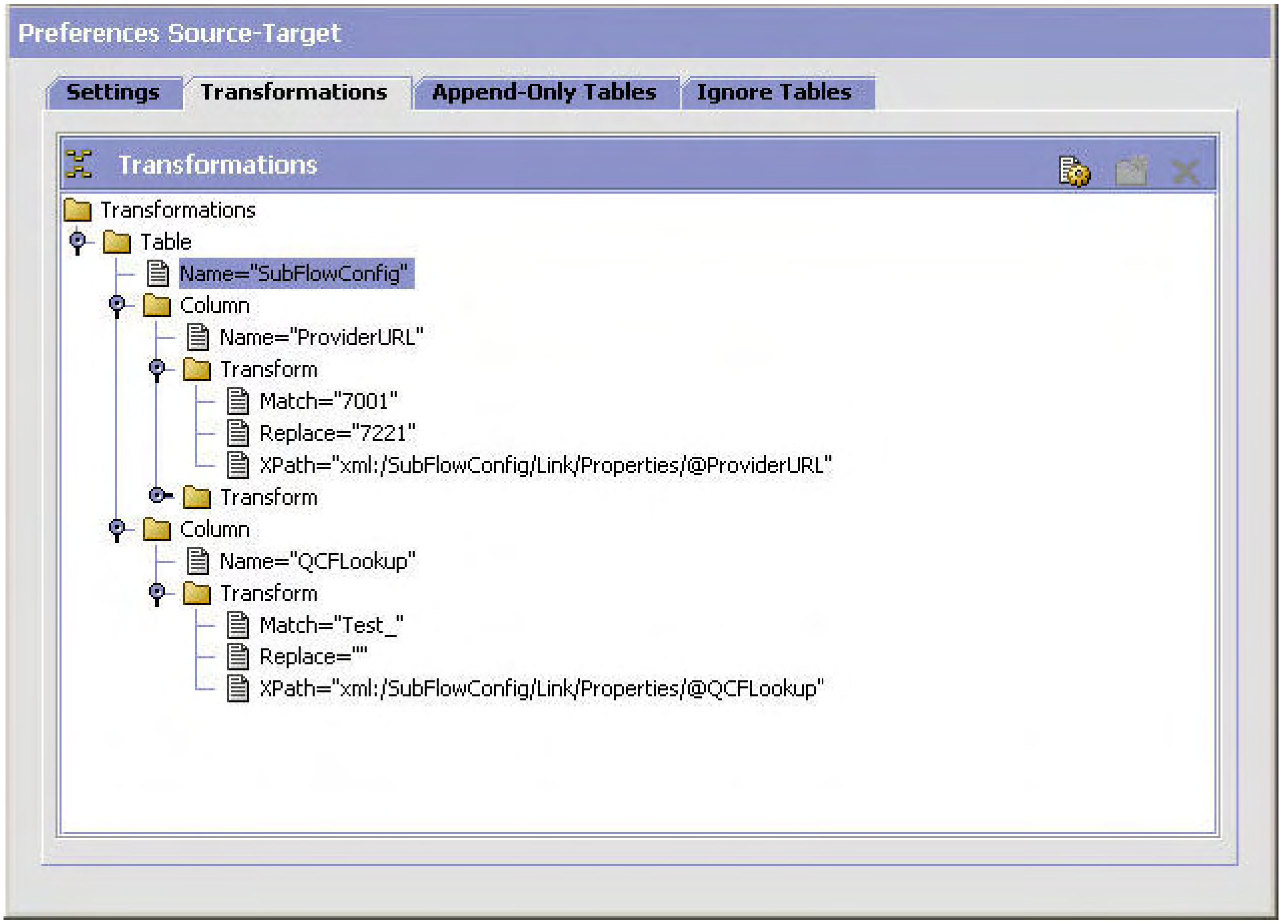
```
<SubFlowConfig>
  <Link>
    <Properties DeliveryMode=""
     InitialContextFactory="weblogic.jndi.WLInitialContextFactory"
     ProviderURL="t3://localhost:7001" QCFLookup="TEST_AGENT_QCF"
     QName="DefaultAgentQueue" TimeToLive=""/>
   </Link>
</SubFlowConfig>
```

The target database has values for port number as 7221 and QCFLookup as AGENT_QCF, which you do not want overridden by values in the source database. To transform these values, specify the values as described in Table 7.

Table 7. Transforming Elements

Element	Attribute	Value
ProviderURL		
	Match	Specify 7001 (or 0 to find all occurrences of 0).
	Replace	Specify 7221 (or 2 to replace all occurrences of 2).
	XPath	xml:/SubFlowConfig/Link/Properties/@ProviderURL
QCFLookup		
	Match	Specify TEST_ to find all occurrences of TEST_.
	Replace	Leave blank to ensure that TEST_ is removed.
	XPath	xml:/SubFlowConfig/Link/Properties/@QCFLookup

Using this example, the **Transformation** tab would look as shown in Figure 7 on page 32.



```
Transformations
  Table
    Name="SubFlowConfig"
    Column
      Name="ProviderURL"
      Transform
        Match="7001"
        Replace="7221"
        XPath="xml:/SubFlowConfig/Link/Properties/@ProviderURL"
      Transform
    Column
      Name="QCFLookup"
      Transform
        Match="Test_"
        Replace=""
        XPath="xml:/SubFlowConfig/Link/Properties/@QCFLookup"
```

Figure 7. Transformations Example

Transforming Elements Using the CDT

About this task

To transform elements of the configuration data:

Procedure

1. From the Deployment Explorer window action bar, choose the *Preferences* icon.
2. In the Preferences window, select the Transformations tab and fill in values. When you deploy data, these transformation values you specify are deployed along with configuration data.

   Before you deploy data, you must first perform a database comparison as described in "Compare Source and Target Databases for CDT" on page 33.

Chapter 8. Comparing Source and Target Databases for the CDT

Compare Source and Target Databases for CDT

To deploy configuration data into production, compare the two databases, then deploy your changes.

Note: The CDT considers special characters as data when both source and target environment are databases.

After completing the comparison, you can perform any of the following actions:
- Examine any differences, using the instructions in "Examining Database Differences Using the CDT" on page 35.
- Export the report, using the instructions in "Exporting Comparison Results Using the CDT" on page 37.
- Generate a report, using the instructions in "Generating a Report of Differences Using the CDT" on page 37.
- Deploy your changes, using the instructions in "Deploying Configuration Data Using the CDT" on page 41.

Comparing Source and Target Data Using the CDT

About this task

When you are deploying data for the ***first*** time, compare the entire database. Compare smaller increments only ***after*** you are certain that your source and target databases have relatively few differences.

To compare the source and target databases:

Procedure

1. From the Deployment Explorer tree, select the data you want to compare.

   The available comparison groups are as follows:

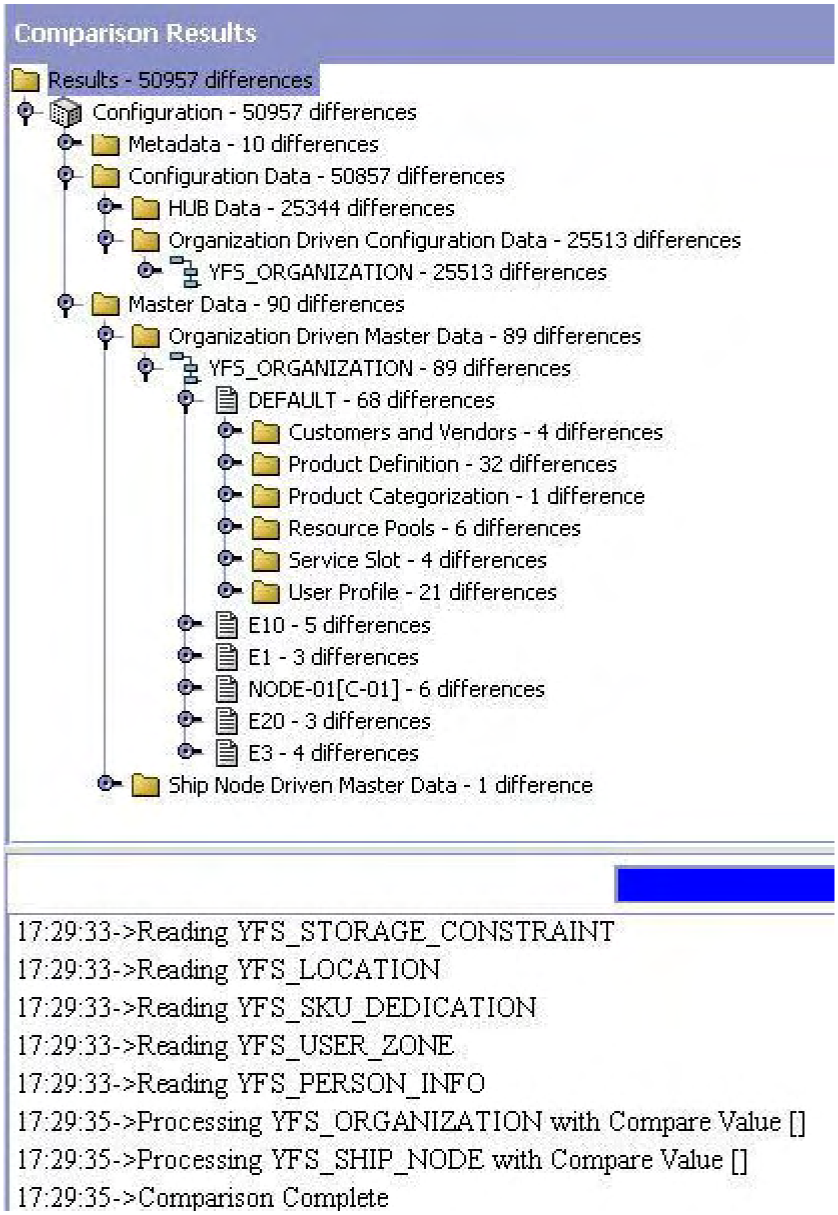
   - For the entire database, choose the *Sterling Selling and Fulfillment Foundation* **Configuration** icon.
   - For a specific configuration group, choose the **Configuration Group** icon.
   - For a specific driver entity, choose the **Driver Entity** icon.
2. From the Deployment Explorer action bar, choose the **Compare** icon. The Comparison Results window displays on the top right and lists all differences. The Comparison Results Status panel displays on the bottom right.

   If you did not specify the `CompareOrganizationCode` argument as part of the ydk command, the CDT prompts you to specify the organizations that you are comparing, as explained in "Specify Organizations to Compare Using the CDT" on page 34.

   Additionally, if you comparing data in a multischema deployment, the CDT prompts for a Colony ID, as explained in "About Comparing Data in a Multischema Environment Using the CDT" on page 35.

Results

Note: If a table is present in multiple groups or under multiple entities, its difference may be counted multiple times in the total number.



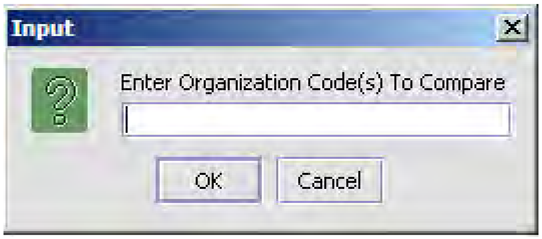
```
Results - 50957 differences
  Configuration - 50957 differences
    Metadata - 10 differences
    Configuration Data - 50857 differences
      HUB Data - 25344 differences
      Organization Driven Configuration Data - 25513 differences
        YFS_ORGANIZATION - 25513 differences
    Master Data - 90 differences
      Organization Driven Master Data - 89 differences
        YFS_ORGANIZATION - 89 differences
          DEFAULT - 68 differences
            Customers and Vendors - 4 differences
            Product Definition - 32 differences
            Product Categorization - 1 difference
            Resource Pools - 6 differences
            Service Slot - 4 differences
            User Profile - 21 differences
          E10 - 5 differences
          E1 - 3 differences
          NODE-01[C-01] - 6 differences
          E20 - 3 differences
          E3 - 4 differences
      Ship Node Driven Master Data - 1 difference
```

```
17:29:33->Reading YFS_STORAGE_CONSTRAINT
17:29:33->Reading YFS_LOCATION
17:29:33->Reading YFS_SKU_DEDICATION
17:29:33->Reading YFS_USER_ZONE
17:29:33->Reading YFS_PERSON_INFO
17:29:35->Processing YFS_ORGANIZATION with Compare Value []
17:29:35->Processing YFS_SHIP_NODE with Compare Value []
17:29:35->Comparison Complete
```

Figure 8. Comparison Results Window

Specify Organizations to Compare Using the CDT

If you did not pass the `CompareOrganizationCode` argument as part of the ydk command, the CDT displays the Input pop-up window when you choose the **Compare** icon in the Deployment Explorer window. The Input pop-up window prompts you to specify the organizations that you want to compare.

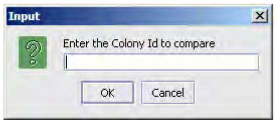


Enter Organization Code(s) To Compare

Use a comma-delineated format to specify organizations to compare, such as Org-1,Org-2. If you click OK without entering an organization, the CDT compares all organizations. It is recommended that all participating organizations are compared together. For example, if two organizations, Org-01 and Org-02, participate with two ship nodes, Node-01 and Node-02, you should specify Org-1,Org-2,Node-01,Node-02.

About Comparing Data in a Multischema Environment Using the CDT

If you are comparing data in a multischema environment, the CDT displays a prompt for the Colony ID to compare:



Enter the Colony Id to compare

If you click OK without entering a Colony ID, the CDT compares all colonies.

When you enter a Colony ID and click OK, the CDT compares data only for that colony, which is the set of organizations that belong to the colony. If you have other colonies in your configuration schema that you want to compare, you must compare each colony separately. For example, you must compare Colony_01 and Colony_02 in separate comparisons.

If you are comparing data in a master data schema, CDT displays the Input pop-up window instead of displaying a prompt for the Colony ID. This is because the master data is the data of an entire colony.

When you enter the name of an organization in the Input pop-up window, CDT compares only the data of the organization that belongs to the particular colony.

If you click OK without entering an organization in the Input pop-up window, CDT compares all the colonies.

Examining Database Differences Using the CDT

About this task

To examine the differences between databases:

Procedure

1. Choose to compare the two databases. For more information about comparing databases, see “Comparing Source and Target Data Using the CDT” on page 33.
2. From the Comparison Results tree, expand the corresponding entity and select the table that you want to examine.

3. From the Comparison Results action bar, choose . The range of possible results are as follows:

   - The **Unchanged** icon indicates that an entity contains dependent tables that have differences.

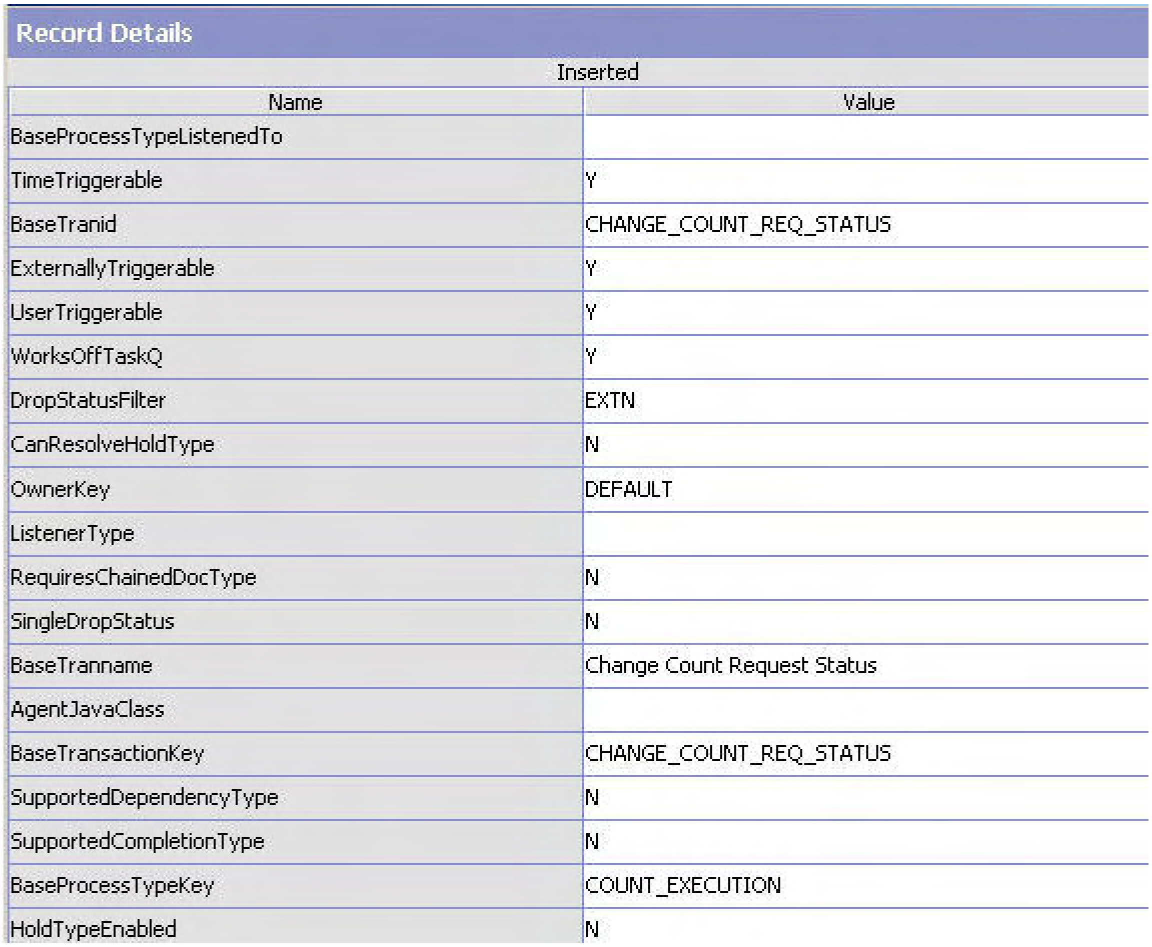
   - The **Add** icon indicates that a record is inserted to the target database.

   - The **Remove** icon indicates that a record is deleted from the target database.

   - The **Modify** icon indicates columns that are updated on the target database.

4. Choose an entity's icon, as described in step 3, to display the Record Details window for the entity. The window displays data about the entity's values in the target database. For example, the Record Details window for modified data displays the following sections:
   - Top section - displays values that are changed on the target database
   - Bottom section - displays values that remain the same.

   Figure 9 shows the Record Details window for inserted data.



Record Details

Inserted	
Name	Value
BaseProcessTypeListenedTo	
TimeTriggerable	Y
BaseTranid	CHANGE_COUNT_REQ_STATUS
ExternallyTriggerable	Y
UserTriggerable	Y
WorksOffTaskQ	Y
DropStatusFilter	EXTN
CanResolveHoldType	N
OwnerKey	DEFAULT
ListenerType	
RequiresChainedDocType	N
SingleDropStatus	N
BaseTranname	Change Count Request Status
AgentJavaClass	
BaseTransactionKey	CHANGE_COUNT_REQ_STATUS
SupportedDependencyType	N
SupportedCompletionType	N
BaseProcessTypeKey	COUNT_EXECUTION
HoldTypeEnabled	N

Figure 9. Record Details Window

5. After examining your data, you may want to generate a report of these differences as described in “Generating a Report of Differences Using the CDT.”

Exporting Comparison Results Using the CDT

About this task

You can export the configuration differences for comparison at a later time or as a backup for your existing configuration.

To export the comparison results into an XML file:

Procedure

1. Ensure that you have specified a directory location where the comparison report is generated in the Reports Directory field in the Settings tab of your Sterling Selling and Fulfillment Foundation Configuration Deployment Tool Preferences. For more information about specifying these preferences settings, see “Setting CDT Preference Settings in GUI Mode” on page 22.
2. From the **Comparison Results** action bar, choose 💾. From the Windows Explorer, browse to the location specified in the **Reports Directory** field.

   The Sterling Selling and Fulfillment Foundation Configuration Deployment Tool automatically creates a subdirectory in this directory. For example, if you have specified `D:/reports` in the **Reports Directory** field and exported the comparison results at 3:40 pm on May 23, 2008, CDT creates the subdirectory as: `D:/reports/export20080523154024`. This new subdirectory contains the `ydkexport.xml` file, which contains the comparison results.

   After the import gets completed, you can perform any of the following actions:
   - Examine the differences using the instructions in “Examining Database Differences Using the CDT” on page 35.
   - Generate a report of the differences, using the instructions in “Generating a Report of Differences Using the CDT.”
   - Deploy your changes using the instructions in “Deploy Configuration Data” on page 41.

Generating a Report of Differences Using the CDT

About this task

You can generate a report of differences between the source and target databases.

To generate a report of differences:

Procedure

1. Choose to ensure that you have specified a `reports` directory. For more information about specifying CDT settings, see “Setting CDT Preference Settings in GUI Mode” on page 22.
2. Choose to compare the two databases. For more information about comparing databases, see “Comparing Source and Target Data Using the CDT” on page 33.
3. In the Comparison Results tree, select the Results node.

4. From the **Comparison Results** action bar, choose ▦. The **Status** panel displays trace messages that enable you to determine the success of the report generation process. The location of the report displays along with a message of the successful creation of reports.
5. From the Windows Explorer, browse to your reports directory. Within the directory you specified, the Configuration Deployment Tool creates a subdirectory named according to the time it was created.

   For example, if you have specified `D:/reports` as the reports directory and generate a report at 3:40 p.m. on May 23, 2008, the CDT creates a subdirectory called `20080523154024` within the `D:/reports` directory.

   This new subdirectory contains the following:
   - An `index.xml` file contains an overall summary of changes as displayed on the UI
   - One XML file for each table that has changes with the details of each change
6. Open the XML files to see the differences.

Results

If you generate another report, a new directory is created and populated with another set of XML files.

Importing Configuration Differences Using the CDT

About this task

You can import configuration differences that are obtained by exporting comparison results.

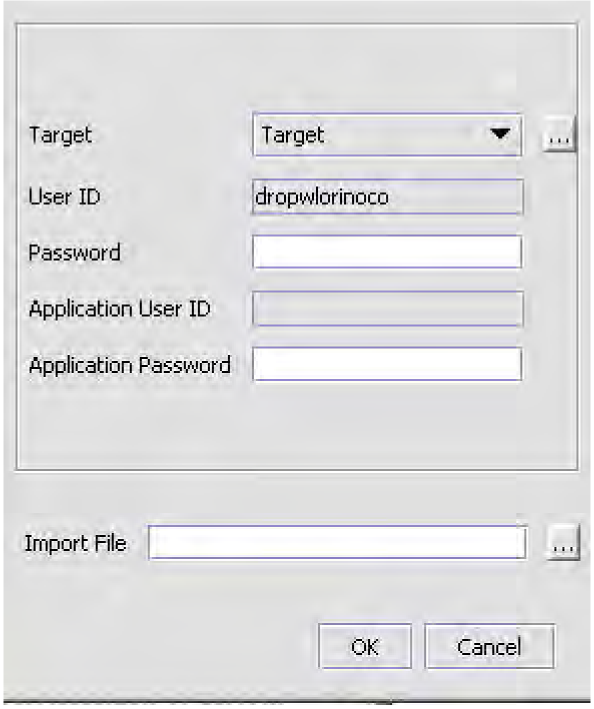
The Configuration Deployment Tool does not support data that contains special characters when comparing databases, exporting comparison results or importing comparison results.

For more information about exporting comparison results, see "Exporting Comparison Results Using the CDT" on page 37.

To import configuration differences:

Procedure

1. Run the `<INSTALL_DIR>/bin/ydk.cmd` script. This script opens a Microsoft Windows console and starts the Sterling Selling and Fulfillment Foundation Development and Deployment WorkBench.
2. From the Sterling Selling and Fulfillment Foundation Development and Deployment WorkBench menu, choose **Tools** → **Deployment** → **Import Results**. The Sterling Selling and Fulfillment Foundation Configuration Deployment Tool Import dialog box displays.



Target: Target

User ID: dropwlorinoco

Figure 10. Import Dialog Box

3. Choose the **Target** button and enter the values appropriate for the Target database. Then choose the **Import File** and enter the path of the file to be imported.

   When you are finished, close the dialog box by clicking **OK**.

4. After the comparison results are loaded, you can perform any of the following actions:
   - Examine any differences, using instructions as specified in "Examining Database Differences Using the CDT" on page 35.
   - Generate a report, using the instructions as specified in "Generating a Report of Differences Using the CDT" on page 37.
   - Deploy your changes, using instructions as specified in "Importing Configuration Differences Using the CDT" on page 38.

Chapter 9. Deploy Configuration Data

Deploy Configuration Data

Before deploying configuration data, ensure that you are deploying the correct data by comparing the data and examining any differences.

In addition, ensure that you have addressed rollback issues. Sterling Selling and Fulfillment Foundation supplies sample backup and rollback scripts. For information on these scripts, see “Data Rollback Scripts” on page 59.

Tip: When you deploy data for the first time, deploy the entire database. Deploy smaller increments only after you are certain that your source and target databases have relatively few differences.

Note: When comparing ship nodes, even if a node in the source schema does not have the adjustment reason code values PACK, SHIP, or RECEIVE, these values will be defaulted to the node in the target schema.

Deploying Configuration Data Using the CDT

About this task

To deploy your configuration data:

Procedure

1. Compare the two databases (by choosing the **Compare** icon). For this detailed procedure, see “Comparing Source and Target Data Using the CDT” on page 33.
2. From the Comparison Results tree, select the entities you want to deploy.

   The available options are as follows:

   - For the entire database, choose the Sterling Selling and Fulfillment Foundation **Configuration** icon. This option is only available if you compared the entire database in step 1.
   - For a specific configuration group, choose the **Configuration Group** icon.
   - For a specific driver entity, choose the *Driver Entity* icon.
   - For a specific record listed under the driver table, choose the icon for that record.
3. From the **Comparison Results** action bar, choose the **Deploy** icon.
   - If the deployment succeeds, the Status panel displays a success message, the data is committed to your target database, and the cache is updated as specified in the `httpurl` field described in Table 3 on page 21.
   - If the deployment fails, the Status panel indicates the errors to resolve and no data is committed to the target database.
4. If you have deployed data from the `YFS_RESOURCE` table, restart your application servers in the target environment in order to refresh the cache.

Audits Generated by the CDT

During the deployment of the configuration data, the audits are generated in the target database. These audits can be used for tracking changes and rolling back the data. These audits are not deployed from the source database, they are generated based on the entity definitions located in the `<INSTALL_DIR>/repository/entity` directory. For more information about Generating Audit References for Entities, see the *Selling and Fulfillment Foundation: Extending the Database Guide*

Driver XML Files Used by the CDT

Sterling Selling and Fulfillment Foundation provides two driver xml files that are used by the CDT to display logical or functional grouping of tables into which data is deployed. These files organize the tables in a hierarchical manner based upon the functional dependencies among them. The actual data deployment is carried out in accordance with this structure or hierarchy. The config-db.xml and master-db.xml files are located in the `<INSTALL_DIR>/database/cdt/` directory.

Deploying Your Configuration Data in Command-Line Mode

About this task

There may be circumstances under which you want to run or schedule deployment of configuration data without user interface interaction or without viewing the source and target comparison results.

To accomplish this you can deploy your configuration data in command-line mode. When you deploy your data in command-line mode, CDT automatically compares the source and target environments and then deploys the configuration data.

To deploy configuration data in command-line mode:

Procedure

1. Set the properties in the `<INSTALL_DIR>/bin/cdtshell.cmd` file for windows (`cdtshell.sh` for UNIX/Linux) as described in Table 8.

Table 8. Configuration Deployment Tool Properties

Property	**Description**
SOURCE_DB	The logical identifier for the source database. This value corresponds to the `Name` setting in the `<SourceDatabases>` element in the ydkprefs.xml file.
SOURCE_PASSWORD	Optional. If using a database as the source destination for data, specify the password for the database instance.
TARGET_DB	The logical identifier for the target database. This value corresponds to the `Name` setting in the `<TargetDatabases>` element in the ydkprefs.xml file.
TARGET_PASSWORD	Optional. If using a database for the target destination, specify the password for the database instance.
TARGET_HTTP_PASSWORD	Specify the password of the http user required to refresh the cache on the server. **Note:** This parameter is not required if there is no cache to refresh on the server.

2. Run the `<INSTALL_DIR>/bin/setupfiles.cmd` script for windows (`setupfiles.sh` for UNIX/Linux).

3. Run the `<INSTALL_DIR>/bin/cdtshell.cmd` script for windows (`cdtshell.sh` for UNIX/Linux).

Results

You can also schedule this script to run at any appropriate time.

You can use the following MODE command line options with CDT:

- MODE Deploy—To deploy changes from source to target database. The various other options available with this MODE argument are:
  - ExportDir <directory>—The specified <directory> will be created, and the results of the comparison are stored in that <directory>.
  - ExportPassphrase <password>—The specified password will be used to encrypt supported Import or Export data when exporting results. This is only applicable when ExportDir parameter is passed.
  - ImportDir <directory>—The specified <directory> should contain the exported results. Instead of comparing the source and target databases, this export will be loaded. When you pass this argument, the source database properties are not used.
  - ImportPassphrase <password>—The specified password will be used to decrypt data from the import files, and it should match the password given to create the export file.
  - DoNotSynchronize <Y|N>—If you pass Y, only the comparison the is done; nothing is deployed. By default, exported results are automatically deployed.

For example, if you want to perform comparison and export the comparison results to specific directory. But you do not want to deploy those changes, pass the following arguments:

```
-MODE Deploy -ExportDir C:\CDT\Reports -DoNotSynchronize Y
```

If you do not pass the MODE java argument explicitly, CDT uses MODE Deploy option as default.

- MODE CDT2IE—To convert the CDT comparison export format to the Import or Export file format. The two required parameters that you need to pass with this MODE argument are:
  - InputFile <Path to the `ydkexport.xml` file>
  - OutputFile <Name of the Import or Export file to write>

    For example:

    ```
    -MODE CDT2IE -InputFile C:\CDT\ydkexport.xml -OutputFile NewImportFile.xml
    ```
- MODE IE2CDT—To convert the Import or Export file format to the CDT comparison file format. The two required parameters that you need to pass with this MODE argument are:
  - InputFile <Name of the Import or Export file to write>
  - OutputDir <Path to the directory where the CDT comparison file should be stored>

    For example:

    ```
    -MODE IE2CDT -InputFile MyImportFile.xml -OutputDir C:\CDT\Reports
    ```
- MODE LABELDEPLOY—To deploy audit changes between two version labels. The two available options with this MODE argument are:
  - FromLabel <Label identifier to start from>
  - ToLabel <Label identifier to end with>

For example:

```
-MODE LABELDEPLOY -FromLabel MyLabel1 -ToLabel MyLabel2
```

Chapter 10. Configuration Data Versioning Tool Overview

Configuration Data Versioning Tool Overview

Configuration data is an integral part of all implementations. Often, there is a need to track changes to an implementation's configuration. Furthermore, if the changes in configuration data are found to be inadequate, there is no easy way of rolling back changes to their original states.

In an offsite and onsite implementation model, master configuration data is maintained onsite, which is where the production environment is hosted. When a fix pack must be applied to the production environment, offsite test developers must write instructions regarding any configuration data changes for the fix pack and pass it to the onsite configuration manager; that is, certain values of business rules need to be changed. The onsite configuration manager has to replicate the configuration changes onto the production environment.

To make it easier to track versions of configuration data or sets of changes to configuration data, Sterling Selling and Fulfillment Foundation includes the Configuration Data Versioning tool (CDV Tool) It enables you to capture changes from a source database, compare and deploy them onto a target database (this can be the same or a different database).

To enable this functionality, the configuration table must have `AuditRequired` flag set to `Y` and the table name must exist in config-db.xml. By default, most of the configuration tables have `AuditRequired` flag set to `Y`.

The CDV Tool supports configuration tables only.

You can create version labels in the Configurator to represent timestamps in a time line when changes occur in configuration data. The system can then identify any changes in the configuration data between timestamps of version labels based on the audit information in the system.

When configuring the audit purges, if you purge the YFS_AUDIT table records from the deployable tables, the Configuration Data Versioning Tool will not deploy those changes.

Configuration Data Versioning Tool Features

The Configuration Data Versioning (CDV) Tool can be accessed from the Sterling Selling and Fulfillment Foundation Development and Deployment WorkBench (also known as the "WorkBench").

The Configuration Data Versioning Tool allows you to select different version labels from a source database, compare the data and apply them to a target database. The system identifies modifications made to the source database between two version labels. You can see the details of each modification and detect conflicts. These modifications are applied to the target database while checking for conflicts. Once all conflicts are resolved, you can deploy the changes.

Version label data is stored in the YFS_CONFIG_VERSION_LABEL table. The manageConfigVersionLabel API and getConfigVersionLabel API are used to manage and search for config version labels.

Configuration Data Version Labels Examples

The following table shows two version labels, VersionLabel_1 and VersionLabel_2, and some samples of their associated changes and timestamps:

Table 9. Version Label Example

Version Label	Audited Changes	Timestamp
	Org1 is created.	5/1/2007 9:00
	Org2 is created.	5/1/2007 9:10
VersionLabel_1		5/1/2007 9:20
	Org1's DefaultFulfillmentType is updated from "A" to "B"	5/1/2007 9:30
	Org1's DefaultFulfillmentType is updated from "B" to "C"	5/1/2007 9:40
	Org2 is deleted	5/1/2007 9:50
	Org3 is created	5/1/2007 10:00
VersionLabel_2		5/1/2007 10:10
	Org4 is created	5/1/2007 10:20
		Current Timestamp

You can obtain the changes based on the configured version labels in the Sterling Development and Deployment WorkBench, including both forward and backward changes.

Forward Changes

These are changes from an earlier version label to a later version label (or the latest changes).

In this case, it can be from VersionLabel_1 to VersionLabel_2 or Version Label_1 to current timestamp. This can be used in the scenario where users want to capture the certain changes from one system and deploy those changes onto another system.

Forward Changes Example

Using the Version Labels table as the basis for this example, if the user wants to obtain changes from VersionLabel_1 to VersionLabel_2, the following changes will be captured to be applied later on another system:

Table 10. Forward Changes Example

Changes to be Applied
Update Org1's DefaultFulfillmentType from “A” to “C”
Delete Org2
Create Org3

Backward Changes

These are changes from the current time to an earlier version label.

The audit data will be translated in the **reverse logic** as the changes to be applied later.

Backward Changes Example

Using the Version Labels table as the basis for this example, if a user wants to obtain all changes from the current timestamp to VersionLabel_1 so that those changes can be used in rolling back all changes modified after VersionLabel_1. The following changes will be obtained.

Table 11. Backward Changes Example

Header
Delete Org4
Delete Org3
Create Org2
Update Org1's DefaultFulfillmentType from “C” to “A”

About Setting CDV Tool Preferences

You can configure preferences (such as a Reports Directory) and parameters for resolving conflicts for the Configuration Data Versioning.

Before you can use the comparison tools, you must create version labels for the databases using the Applications Manager. For more information about creating configuration version labels, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide.*

Conflict Handling

You can enable conflict handling by selecting **Conflict Handling for Version Deployment** in Preferences. This is used to prevent overriding of any subsequent changes that occur on the target database.

For example, TaxPayerID has changed from “A” to “B” based on the audit information on the source database. However, on the target database the TaxPayerID has been changed from “A” to “C.” When the change is obtained from the source database and compared with the current value in the target database, the system will detect that the value is no longer “A” and present that to the user for validation.

There are three levels of enforcement:

- Validate Old Values: When enabled, the system will detect conflict based on the expected old value of an attribute of a record of the obtained changes from the source database vs. the current value of the corresponding record in the target database. The details screen displays the conflicting attributes.
- Validate Lockid: When enabled, the system detects conflict based on the expected Lockid of a record of the obtained changes from the source database vs. the current Lockid of the corresponding record in the target database. When a conflict is detected, you cannot deploy the changes. If this is selected, the following occurs:
  - The Details screen will display this highlighted text: Conflicting Lockid – Deploy Action is disabled.
  - The conflicting attributes with the Expected Old Values will be displayed.

- The Deploy action will be disabled until all conflicts are resolved.

  Lockid validation is not available for the audit records created before the Sterling Selling and Fulfillment Foundation Release 8.0.
- Validate Record Exists Before Delete: When enabled, the system will validate the record exists in the target database before it attempts to delete it. If the record is not there, the operation will be marked as a conflict.

Setting Preferences for Configuration Data Versioning Tool

Procedure

1. Start the Configuration Data Versioning Tool. For more information about using the Configuration Data Versioning Tool, see "Running the CDV Tool."
2. Select Tools > Configuration Data Version Deployment.
3. The Logon screen displays. Select the names of the desired source and target databases, and enter the password for each. Click **OK**.
4. The Compare screen displays. Click **Preferences**.

   In the Preferences window, you can define preferences for the Configuration Data Versioning tool. For field value descriptions, see Table 4 on page 23.

Running the CDV Tool

About this task

The Windows console displays WorkBench startup information. Do not close the console while the WorkBench is running. Closing the console closes the tool, and your work is lost.

To start the Configuration Data Versioning Tool:

Procedure

1. Start the Sterling Selling and Fulfillment Foundation Development and Deployment WorkBench. On Microsoft Windows, run the `ydk.cmd` script for Windows (`ydk.sh` for UNIX/Linux) from the `<INSTALL_DIR>\bin` directory.
2. From the Sterling Selling and Fulfillment Foundation Development and Deployment WorkBench menu, choose Tools > Deployment > Configuration Data Version Deployment. This opens the Configuration Data Version Tool Logon dialog box.

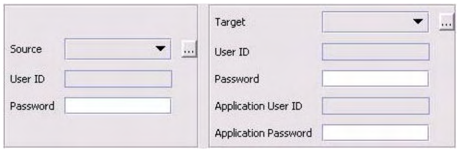


Figure 11. Configuration Data Version Tool Dialog Box

3. Choose the **Source** button and enter the values appropriate for the source database. Then choose the **Target** button and enter the values appropriate for the target database.

   When you are finished, close the dialog box. The values you specified are saved automatically and persist from one session to the next.

If you change the name of either source or target database, the transformation settings are lost. To get back your old transformation settings, revert to the old source and target database name.

In the Source database and Target database windows, specify the applicable values as described in Table 12.

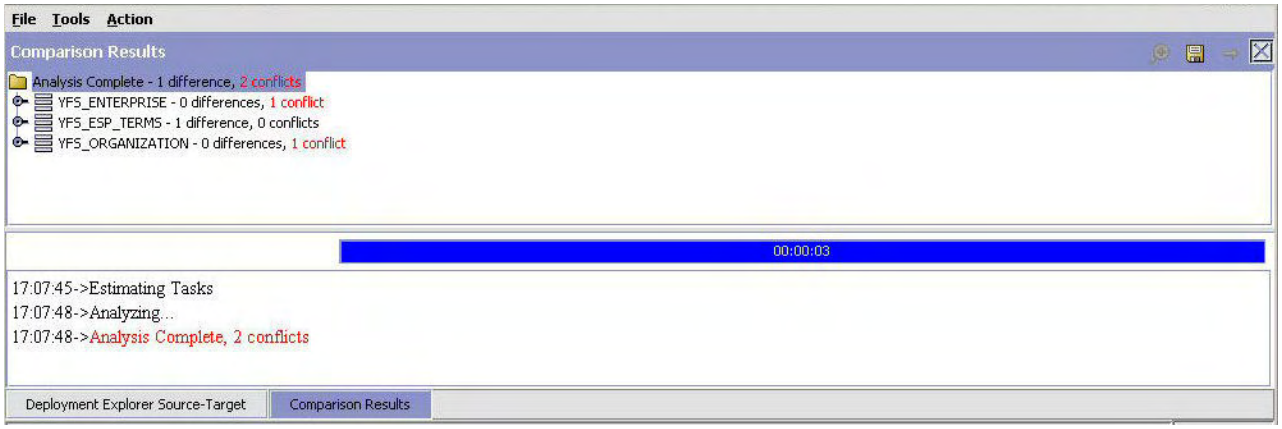
Table 12. Configuration Deployment Tool Logon Dialog Box

Field	Detail
Name	Specify a logical database identifier. For the source, specify the database you want to copy data from. For the target, specify the database to write the data to.
className	Specify the class name of your database driver as follows: • If using Oracle, set to: `oracle.jdbc.OracleDriver` • If using Microsoft SQL Server 2005/2008, set to: `com.microsoft.sqlserver.jdbc.SQLServerDriver` • If using DB2, set to: `com.ibm.db2.jcc.DB2Driver`
jdbcURL	Specify the URL to connect to the database: • If using Oracle, set to: `jdbc:oracle:thin:@<DatabaseServerHostname/IPaddress><TNSListenerPortNumber>:<DatabaseSID>.` • If using Microsoft SQL Server 2005/2008, set to: `jdbc:sqlserver://<Database Server Hostname>:<Port Number>;DatabaseName=<Database name>`. • If using DB2, set to: `jdbc:db2://<Database Server Hostname>:<Port Number>/<Database name>.`
dbType	Specify the type of database you are running. Enter it in all lower case, as shown: • For Oracle, specify `oracle` • For Microsoft SQL Server, specify `sqlserver` • For DB2, specify `db2` • For an XML datasource, specify `xml`
folder	If using an XML datasource, specify the complete path of the folder location for the XML files.
httpurl	Only applicable for the target database. Specify a URL for the application server whose data cache is to be refreshed after data is deployed into the target database. Use the syntax: `http://<hostname/ip-address>:<port-number>/<Application>/interop/InteropHttpServlet`, where `<hostname/IP-address>` is the server address where the application is running, `<port-number>` is the port on which the application is running, and `<Application>` is the name of the application; for example, `smcfs`.
schema	Specify the schema owner as follows: • If you are using Oracle or DB2 database, and the user you specify is different from the Sterling Selling and Fulfillment Foundation schema owner, specify the owner of the Sterling Selling and Fulfillment Foundation schema. • If you are using Microsoft SQL Server, leave this blank.
user	Specify the user name associated with the database.

4. In the Logon dialog box, enter the passwords associated with the user names and Click **OK**.

5. The Compare screen displays. Select a **From Version Label** and a **To Version Label** and click **OK**.

   The system compares the two database versions. Once the comparison is complete, the system displays the Comparison Results screen, as shown in Figure 12.



Analysis Complete - 1 difference, 2 conflicts
YFS_ENTERPRISE - 0 differences, 1 conflict
YFS_ESP_TERMS - 1 difference, 0 conflicts
YFS_ORGANIZATION - 0 differences, 1 conflict

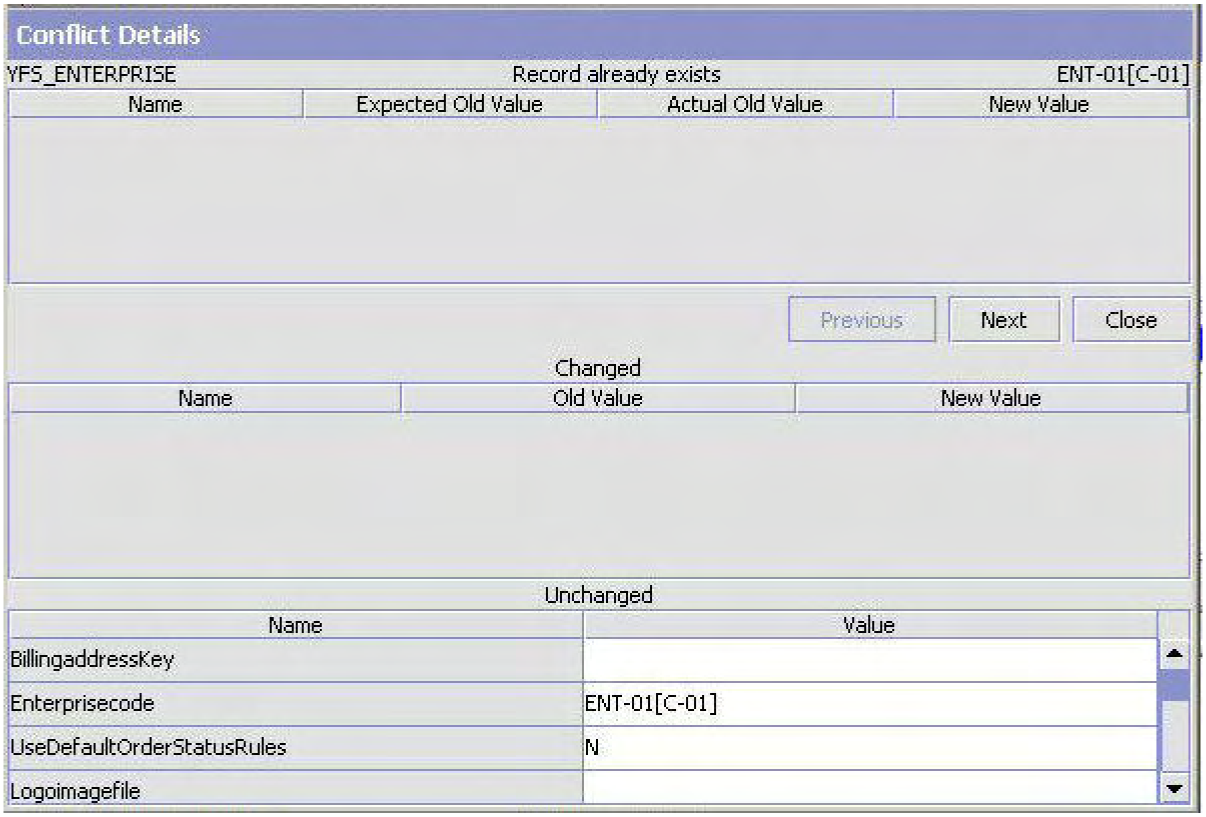
```
00:00:03
17:07:45->Estimating Tasks
17:07:48->Analyzing...
17:07:48->Analysis Complete, 2 conflicts
```

Figure 12. Comparison Results Screen

   Note that the results are grouped by table name onscreen.

6. To see the details for any record included in the results, right-click and select **Details**.

   Use the Applications Manager to create Version Labels. For more information on how to create Version Labels, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.



YFS_ENTERPRISE Record already exists ENT-01[C-01]

Name	Expected Old Value	Actual Old Value	New Value

Changed

Name	Old Value	New Value

Unchanged

Name	Value
BillingaddressKey	
Enterprisecode	ENT-01[C-01]
UseDefaultOrderStatusRules	N
Logoimagefile	

Figure 13. Conflict Details Screen

Table 13. Configuration Deployment Versioning Tool Conflict Details

Field	**Detail**
The first row displays the name of the entity which contains the conflict (for example, YFS_ORGANIZATION), type of the conflict (for example, Record has been modified), and user key (for example, DEFAULT).	
Name	Displays the name of the attribute whose values caused the conflict.
Expected Old Value	Displays the old value that is expected in the target database.
Actual Old Value	Displays the old value that actually exists in the target database.
New Value	Displays the new value that is obtained due to the change in the source database.
Previous	Click this button to view the previous record.
Next	Click this button to view the next record.
Close	Click this button to close the Conflict Details window.
Changed	Displays the name of the changed attribute along with it's old value and new value.
Unchanged	Displays the name of the unchanged attribute along with its existing value.

There are three types of changes that can be displayed:

- Modifications: The Record Details screen displays with a list of changed and unchanged attributes. For changed attributes, the old value and the new

value will be displayed. (The old value is the existing value on the target database; new value is the value of the obtained change from the source database.)
- Insertions: The Record Details screen displays with a list of attributes and their values to be inserted.
- Deletions: The Record Details screen displays with a list of attributes of the entity.

You can scroll through the records by using the **Previous** and Next buttons. When you are done reviewing the details for a record, click **Close**.

Exporting Results from the Configuration Data Versioning Tool

About this task

Before resolving conflicts in the database, you must export the changes to a file so that they can be later deployed on another environment by using the Export Results option. Exporting results for research is only a "must" when conflicts do exist. If there are no conflicts, exporting results is not required.

To export the results to a file:

Procedure

1. Start the Sterling Development and Deployment WorkBench.
2. Select **Tools** → **Configuration Data Version Deployment**.
3. The Logon screen displays. Select the names of the desired source and target databases, and enter the password for each. Click **OK**.
4. The Compare screen displays. Select a **From Version Label** and a **To Version Label** and click **OK**.
5. From the Comparison Results action bar, choose the ⇨ **Deploy** icon.

Results

This option does the following:

- Creates a new folder under the Reports Directory configured in the Preferences with the naming convention `export_<FromVersionLabelID>_<ToVersionLabelID>_<CurrentTS>`
- Creates a version audit file named as `ydkversionexport.xml` file.

Resolving Database Conflicts Using the Configuration Data Versioning Tool

About this task

The database conflicts can only be resolved by editing the version audit file `ydkversionexport.xml`. The `ydkversionexport.xml` file contains all the changes that are applied to the system. Any detected conflict is described within a <Conflict> element. The <Conflict> element contains the information such as conflict type and conflict description.

To resolve conflicts:

Procedure

1. Export the results to the ydkversionexport.xml file (if not already done). For more information about exporting results, see “Exporting Results from the Configuration Data Versioning Tool” on page 52.
2. Open the ydkversionexport.xml file and search for the <Conflict> element to find out the type of conflict. The CDV Tool supports following types of conflict:
   - RECORD_ALREADY_EXISTS
   - RECORD_DOES_NOT_EXIST
   - RECORD_CHANGED

   Below is a sample <Conflict> element entry from the ydkversionexport.xml file:

   ```
   <Conflict ConflictDescription="Record does not exist"
           ConflictType="RECORD_DOES_NOT_EXIST"/>
   ```

How to Resolve Different Types of Conflicts

Resolving RECORD_ALREADY_EXISTS type conflict

This type of conflict happens when CDV tries to perform an Insert operation on a record which already exists in the database. You can resolve this type of conflict in any of the following ways:

- Remove the entire <Audit> element to suppress the change.
- Modify the <Audit> element and make it an Update operation instead of an Insert operation. This can be done by searching for the FoundOldValue attribute within the <ConflictAttribute> element and setting it to match the value mentioned in the <OldValue> attribute within the <Audit> element. After this, change the value of the Operation attribute within the <Audit> element from Insert to Update.

Resolving RECORD_DOES_NOT_EXIST type conflict

This type of conflict happens when CDV tries to perform an Delete or Update operation on a record which does not exist in the database. You can resolve this type of conflict in any of the following ways:

- Remove the entire <Audit> element to suppress the change.
- Create the expected record in the target database. This can be done with the help of Sterling Selling and Fulfillment Foundation UI or API, or running the CDT.

Resolving RECORD_CHANGED type conflict

This type of conflict happens when CDV tries to perform an Update operation on a record whose value has already changed in the database. You can resolve this type of conflict in any of the following ways:

- Remove the entire <Audit> element to suppress the change.
- Create or modify the value of OldValue attribute to match the value of FoundOldValue attribute within the <ConflictAttribute> element and change the value of NewValue attribute if you want to change the value.

  Below is a sample <Audit> element entry from the ydkversionexport.xml file:

  ```
     <Audit AuditKey="20081219134644594514" Operation="Insert"
     TableKey="20081219134644594512" TableName="YFS_QUEUE_SUBSCRIPTION">
     <AuditDetail AuditType="QueueSubscription">
     <IDs>
     <ID DataType="class java.lang.String" Name="QueueKey"
  Value="20081110160649488514"/>
  ```

```
   </IDs>
   <Attributes>
   <Attribute DataType="class com.yantra.yfc.date.YTimestamp"
   Name="Createts" NewValue="20081219134644"/>
   <Attribute DataType="class com.yantra.yfc.date.YTimestamp"
   Name="Modifyts" NewValue="20081219134644"/>
   <Attribute DataType="class java.lang.String" Name="UserKey"
NewValue="20051121111734438480"/>
   </Attributes>
   </AuditDetail>
   <Conflict ConflictDescription="Record already exists"
ConflictType="RECORD_ALREADY_EXISTS">
   <ConflictAttribute ExpectedOldValue=""
   FoundOldValue="20081219134644594512"
   Name="QueueSubscriptionKey" NewValue="20081219134644594512"/>
   <ConflictAttribute ExpectedOldValue="" FoundOldValue=""
   Name="Createts" NewValue="2008-12-19T13:46:44+00:00"/>
   <ConflictAttribute ExpectedOldValue="" FoundOldValue=""
   Name="Modifyts" NewValue="2008-12-19T13:46:44+00:00"/>
   </Conflict>
   </Audit>
```

Importing Version Audit File Generated by CDV Tool

About this task

After resolving the conflicts, you can import the version audit file (`ydkversionexport.xml`).

To import the Version Audit File:

Procedure

1. Start the Sterling Development and Deployment WorkBench.
2. Select Tools > Deployment > Import Version Audits.
3. The Logon screen displays. Select the name of the target database, and enter the password for the target database. Browse to the version audit file (`ydkversionexport.xml`), and click **OK**.

Deploying Database Changes Using the CDV Tool

About this task

Before you can deploy changes from one database to another, you must first verify whether there are conflicts. If there are conflicts, you must export the results to a file in order to research and resolve conflicts. Once the conflicts have been resolved, you can re-import the version audit file (`ydkversionexport.xml`) and then deploy the changes to the target database. For more information about resolving conflicts, see "Resolving Database Conflicts Using the Configuration Data Versioning Tool" on page 52.

To deploy the database changes:

Procedure

1. Start the Sterling Development and Deployment WorkBench.
2. Select **Tools** → **Configuration Data Version Deployment**. The Logon screen displays.
3. Select the names of the source and target databases, and enter the password for each. Click **OK**. The Compare screen displays.
4. Select a **From Version Label** and a **To Version Label**, and click **OK**.

5. From the Comparison Results action bar, choose the ⇨ **Deploy** icon. The Input window displays.
6. In Enter a label for this deployment field, enter the label number from which you want to deploy the configuration data.
   - If the deployment succeeds, the Status panel displays a success message. The data is committed to your target database, and the cache is updated as specified in the `httpurl` field described in Table 3 on page 21.
   - If the deployment fails, the Status panel specifies the errors to resolve, and data is not committed to the target database.
   - In a multischema environment, ensure that you run the Colony Map Synchronizer agent after deploying configuration data.

Rolling Back Changes Using the CDV Tool

About this task

You can rollback changes to a given version label. The CDV Tool audits all the records from a label to the current timestamp, figures out the changes, reverses those changes, and applies it back to the database.

To rollback changes to a particular label:

Procedure

1. Start the Sterling Development and Deployment WorkBench.
2. Select Tools > Deployment > Rollback Data To Version Label.
3. Select the name of the database and enter the password. Click **OK**.
4. Select a **To Version Label**, and click **OK**. The Compare screen displays.
5. From the Comparison Results action bar, choose the ⇨ **Deploy** icon. The Input window displays.
6. In Enter a label for this deployment field, enter the label number from which you want to deploy the configuration data.
   - If the deployment succeeds, the Status panel displays a success message. The data is committed to your target database, and the cache is updated as specified in the `httpurl` field described in Table 3 on page 21.
   - If the deployment fails, the Status panel specifies the errors to resolve, and data is not committed to the target database.

Chapter 11. Configuration Deployment Tool Troubleshooting

CDT Troubleshooting

During operations, the Configuration Deployment Tool displays messages in the Status panel that enable you to understand the status of each operation. These messages can be classified as:

- Status
- Warnings
- Unexpected errors

"Configuration Deployment Tool Messages" describes various messages that you may encounter and any relevant corrective actions you should take.

Configuration Deployment Tool Messages

While using the Configuration Deployment Tool, you may encounter either informational messages or warning messages. These message types are described below.

Informational Messages

Informational messages represent the status of the operation being performed. These messages are displayed in the default color (typically black) in the Status panel. Examples of informational messages include:

- Refreshing database cache
- Deployment operation started
- Reading table YFS_ORGANIZATION

Warning Messages

Warning messages typically require corrective action. They are displayed in red on the Status panel. CDT may produce the warning messages described in this section.

WARNING - FK check failed for table <name> to <name2>

This warning message typically indicates that the configuration data that you are trying to deploy causes inconsistent data in the target database.

To analyze and correct this problem:

1. Determine the size of the data set you are deploying. This error typically occurs when trying to deploy a very small set of data, such as only a driver entity or a configuration group. For example, when deploying a pipeline, this error results if the document type to which the pipeline belongs has not been picked for deployment.

   Try resolving this error by selecting a larger set to deploy. For example, instead of deploying a record, deploy the entire group, if possible.
2. If you still encounter this error for a group or you must only deploy a particular record, try synchronizing the foreign table before deploying the data.

3. Occasionally, inconsistent data in the source database causes this error. If this is the case, you must correct the source of the inconsistency before you proceed.

WARNING - Cache Refresh Failed

This error indicates that CDT was unable to inform the application server cluster on the target environment about the newly deployed configuration changes. The reason the cache refresh failed is displayed on the Status panel.

To analyze and correct this problem:

1. Verify the URL specified in the `httpurl` field for the target database. The `httpurl` is accessible from the Logon dialog box. Ensure that the `httpurl` points to a running instance of the application server and has the following format:

   `http://<hostname/ip-address>:<port-number>/smcfs/interop/InteropHttpServlet` where hostname, ip-address and port-number are the parameters used to connect to the application server.
2. If your target environment is not running, no action is required. Sterling Selling and Fulfillment Foundation automatically reads the latest configuration data when it is started.
3. If the target environment is running, you must manually drop the stale database cache using the Sterling Selling and Fulfillment Foundation System Administration Console. Not performing this step may result in Sterling Selling and Fulfillment Foundation not recognizing the changed configuration.

WARNING - The program detected a few abandoned records in the target database.

In most cases, the abandoned records are harmless and do not lead to incorrect operation of Sterling Selling and Fulfillment Foundation. By default, the CDT leaves them untouched.

This warning typically occurs as a result of the following circumstances which are described in detail in “Importing Externally Maintained Configuration Data” on page 7.

- When the CDT determines that records do not belong to a valid driver entity (for example, a pipeline for a process type that no longer exists).
- When the CDT has been configured to ignore certain tables without ignoring all dependent tables.

To analyze and correct this problem:

1. Add the `-DShowAbandoned=Y` Java parameter to the `ydk.cmd` script.
2. Run the `ydk.cmd` script. If the CDT finds abandoned records, it dynamically creates a group called "Abandoned Records" and displays them in the Comparison Results window.
3. Examine these records, and then either ignore them or delete them from the target.

Unexpected Error Messages in the Configuration Deployment Tool

Depending on the severity, messages about unexpected errors are displayed in either of the following places:

- In the CDT Status panel (in red)

- In the Microsoft Windows console used to launch the Configuration Deployment Tool

To analyze and correct these errors:

- If the error indicates an out-of-memory condition, try your previous operation with a smaller set of data.
- Verify that your system specifications comply with the recommendations described in the *Sterling Selling and Fulfillment Foundation: System Requirements Guide*.
- To increase the memory available for the Configuration Deployment Tool, edit the `<INSTALL_DIR>/bin/tmp.cmd` file and set the `-ms` argument value to a higher value. By default, in the `tmp.cmd` file the HEAP_FLAGS Java parameter is set as following.

  ```
  set HEAP_FLAGS=-Xms768m -Xmx768m
  ```

For example, if you were comparing the complete configuration, try comparing one group at a time. The same is true for the deployment operation.

In other cases, the underlying error and detailed trace are displayed. This may point to an incomplete or faulty installation or incorrectly specified runtime parameters.

Exceptions While Exporting Configuration Data with cdtshell Scripts

The cdtshell.cmd (or .sh) scripts throw a java.lang.StringIndexOutOfBoundsException when exporting configuration data using the CDT with the database as SOURCE_DB and the XML file as TARGET_DB.

To analyze and correct this exception:

Verify that the ExportDir and the folder location of the XML files are not the same location.

Data Rollback Scripts

Before deploying data from a staging to a production environment, it is recommended to take a snapshot of your production configuration data. This snapshot enables you to perform a rollback of the deployment operation in case of failure. Sterling Selling and Fulfillment Foundation provides the following rollback scripts:

- Backup script - Creates multiple files containing data from the Sterling Selling and Fulfillment Foundation configuration data.
- Restore script - Uses the files created by the backup scripts to restore the Sterling Selling and Fulfillment Foundation configuration to a previously known good state.

The `<INSTALL_DIR>/bin/backupScriptGen.xml` script generates backup and restore scripts. For more information about running the data rollback scripts, see the *Selling and Fulfillment Foundation: Installation Guide*.

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 2011. Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. 2011.

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

Trademarks

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

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

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

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

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

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

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

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

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

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

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

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

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

Index

Printed in USA