Selling and Fulfillment Foundation

Multitenant Enterprise Guide

Release 9.0

March 2010

Sterling Commerce
An IBM Company

Contents

Introduction to Multitenant Configurations 5
Single-Instance Single-schema Deployment 6
Multiple-Instance Deployment 8
    Multiple Context-Root Deployment 9
Single-Instance Multischema Deployment 11
Multischema Overview 13
    Colonies 13
    Multischema Entity Tables 14
    Facts 15
    Migrate Data in a Multischema Deployment 16
    Sterling Analytics Reporting in a Multischema Deployment 17
Deployment Benefit Matrix 18
Move from Single-Schema Mode to Multischema Mode 20
Add an Enterprise to a Colony 21
Introduction to Managing Colonies 22
Add a Colony Using the GUI Wizard in Windows, UNIX, or Linux 22
Add a Colony from Command-Line Interface (Windows, Linux, or UNIX) 24
Add a Colony in Silent Install Mode 25
    Sample Silent AddColony File 27

Index **29**

Introduction to Multitenant Configurations

As a customer of Selling and Fulfillment Foundation, multitenant configuration options enable you to configure your enterprises in a variety of ways:

- Single-Instance Single-schema Deployment
- Multiple-Instance Deployment
- Single-Instance Multischema Deployment

Each of these options provides benefits and trade-offs according to your individual business requirements. You can opt for a single-instance deployment if you do not anticipate a significant increase in the number of your enterprises and if you want a central point of control and monitoring across your deployment. If your enterprises begin to expand and outgrow a single instance, you can deploy multiple instances. Some business requirements might drive the need for more advanced deployment models where multiple entities require different upgrade cycles and isolation levels. These businesses might include third-party logistics (3PL) providers and hosted environments. For these requirements, you could implement a single-instance multischema deployment.

All of the deployments described in this guide are multitenant, which means they consist of multiple enterprises that have unique business needs, such as different process flows, enterprise specific extensions, and rules. The last of these deployments is also multischema, which is a type of multitenant deployment that has multiple transactional database schemas – one per colony.

This chapter contains information about these deployment options as well as their advantages and limitations. A Deployment Benefit Matrix that summarizes the advantages of each deployment is shown in Deployment Benefit Matrix.

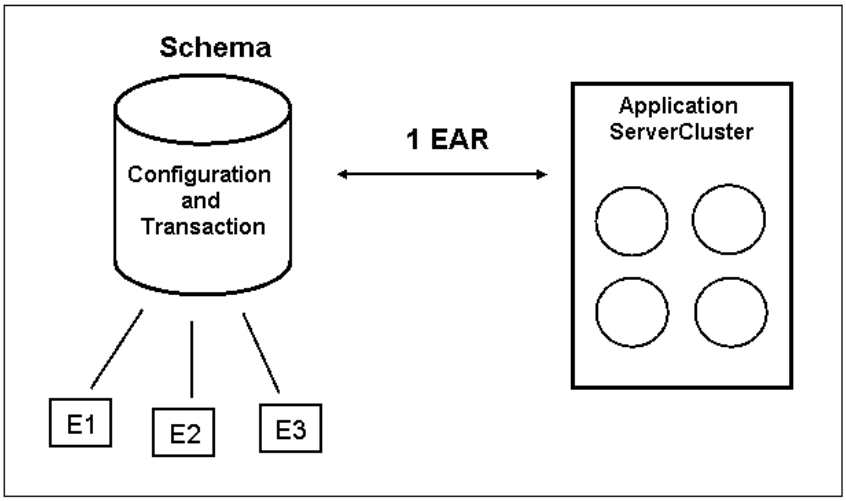
Single-Instance Single-schema Deployment

Selling and Fulfillment Foundation offers multi-tenancy support for single-instance deployments that contain:

- A single database containing all Configuration and Transaction data
- A cluster of application servers with a single Enterprise Archive (EAR)
- One or more enterprises

In this configuration, all extensions can be defined by enterprise. These include code extensions, user exit implementations, events, templates, menus, and extensions to the user interface. Each enterprise can have its own sourcing rules, pipelines, tasks, and so on. In addition, this type of deployment offers isolation of configuration; that is, if one enterprise's configuration changes, the others are not affected. In a single instance, all the data is centrally located, so you can upgrade the entire instance together and consolidate database maintenance tasks.

The following illustration shows an example of single-instance, single-schema deployment:



As this illustration shows, in a single-instance deployment, Configuration and Transaction data reside within the same schema. This schema is shared by three enterprises: E1, E2, and E3. It has a cluster of nodes in an application server and one EAR.

Following are the advantages and limitations of this single-schema deployment.

Advantages:

- Enterprises can participate with each other, sharing inventory and sourcing rules.
- Nodes, such as ship nodes, can be shared across enterprises.
- Enterprises can inherit rules from another enterprise within this same Configuration schema.

- Each enterprise can have its own code extensions, events, templates, extensions to user interfaces, menus, and user exit implementations. If E1 and E2 have different inventory systems, they need to have different interfaces, too. These can all be defined by enterprise.
- Each enterprise in configured separately. For example, E1 can have its own sourcing rules, pipelines, tasks, and so forth. This isolation of configuration means that you can deploy and move data by enterprise in the Configuration Deployment Tool (CDT). For more information about the CDT, see the *Selling and Fulfillment Foundation: Configuration Deployment Tool Guide*.
- All enterprises are under a single point of control. For example, you can start and stop agents and monitor processes from one place.

Limitations:

- In this current deployment, enterprises cannot upgrade independently by enterprise, because Transaction data is not isolated by enterprise. It is in a single-schema.
- As more enterprises are added, more database capacity is needed and increased hardware expense can result.

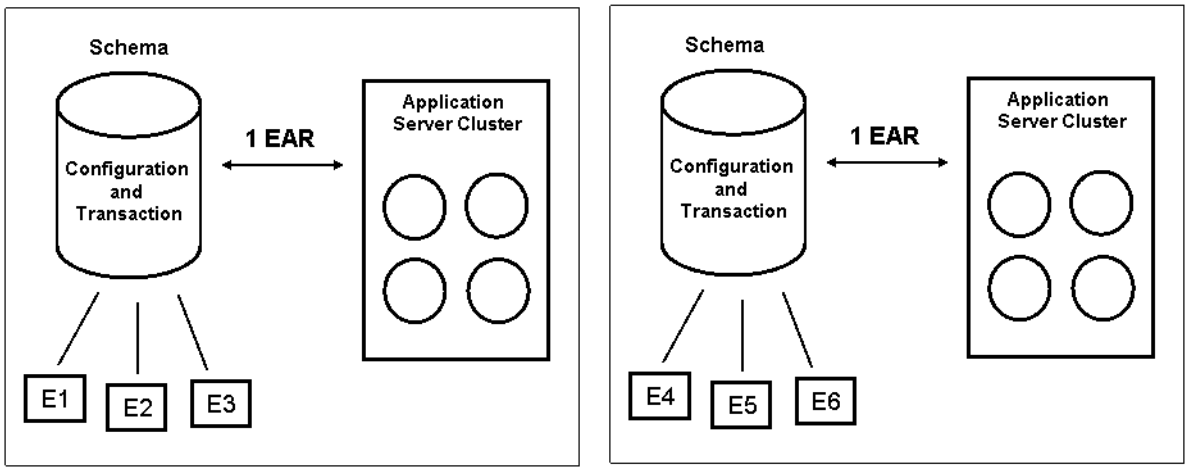
There is a finite threshold for growth in a single-instance deployment. Also, if you want more isolation for your enterprise environments, you can consider a multiple-instance deployment.

Multiple-Instance Deployment

In a multiple instance deployment, you can have two or more instances of databases, application servers, and EARs installed. If database size limitations were an issue in a single-instance deployment, you can keep adding instances in a multiple-instance deployment as the number of your enterprises grows.

A multiple instance deployment can also consist of separate enterprises or groups of enterprises, each with its own context root or WebLogic EAR, as described in Multiple Context-Root Deployment.

The following illustration shows an example of a multiple-instance deployment:



Following are the advantages and limitations of this multiple-instance deployment.

Advantages:

- Enterprises can increase database size and continue to add instances.
- Enterprises can upgrade each instance independently.

Limitations:

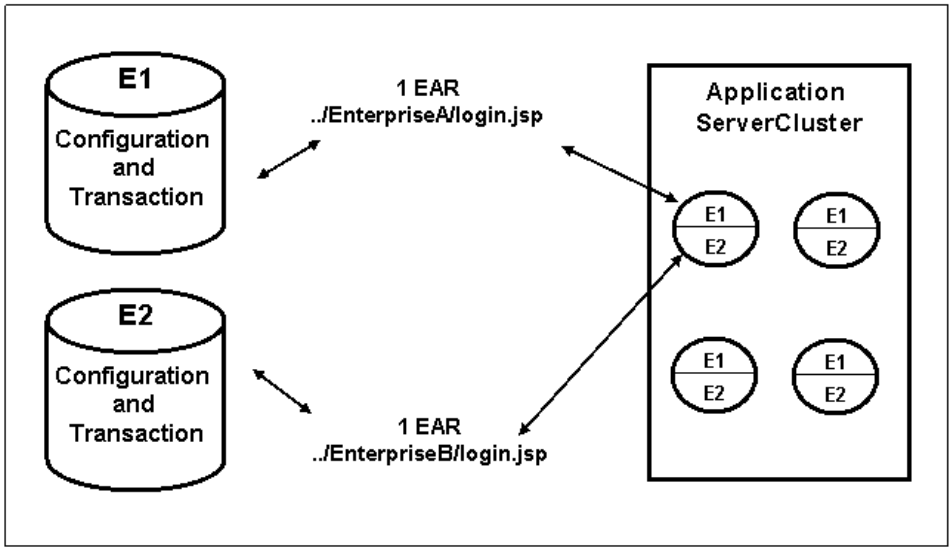
- These instances are not connected to each other and cannot share Configuration data, such as hub-level pipelines. They also might require synchronization of Configuration data.
- There is more than one point of control and monitoring. For example, you need to run and monitor agents separately.

Many multiple-instance deployments enjoy the advantages listed above and are unaffected by the limitations, because they have no need to share data between instances or maintain a central point of control. They benefit from being able to add databases as their enterprises grow.

However, if hardware resource or memory constraints require that you run separate enterprise deployments within each node on the application server, you can achieve this with a multiple context-root deployment.

Multiple Context-Root Deployment

In a multiple context-root deployment, each node on the application server can have separate data for each enterprise or groups of enterprises. As the following illustration shows, this separation of enterprises means that there is a WebLogic EAR for each enterprise (or for a group of enterprises) as well as a customer-specified URL that points to enterprise-specific login screens.



In this scenario, each enterprise has its own context root, or EAR. Enterprise A's data resides in E1, while Enterprise B's data resides in E2. Each enterprise can upgrade independently and can completely separate data from each other because the databases connect only to their respective EARs.

The *Selling and Fulfillment Foundation: Customization Basics Guide* contains information about how to deploy multiple EARs and multiple context roots.

Following are the advantages and limitations of this multiple context-root deployment:

Advantages:

- Each node is a deployment of the Websphere, WebLogic, or the JBoss application server. You can monitor application servers at one, aggregate level.
- You can deploy one more EAR within each application server node. You can have multiple EARs and an enterprise-specific login interface.
- Enterprises can upgrade each instance independently.
- Isolation of context roots can result in better and more dynamic utilization of hardware capacity, especially if you ensure that the enterprises sharing an application server require different peak usage periods.

Limitations:

- No central point of control for E1 and E2, which results in separate monitoring of agents.
- No sharing of data between E1 and E2.

- No gain in memory utilization. For example, because Configuration data is not shared, configuration of resource permissions will be duplicated, resulting in approximately double the memory consumption.

In this deployment, you cannot leverage the same Configuration data across the deployment while still keeping Transaction data separate among enterprises. These kinds of benefits are available in a multischema deployment.

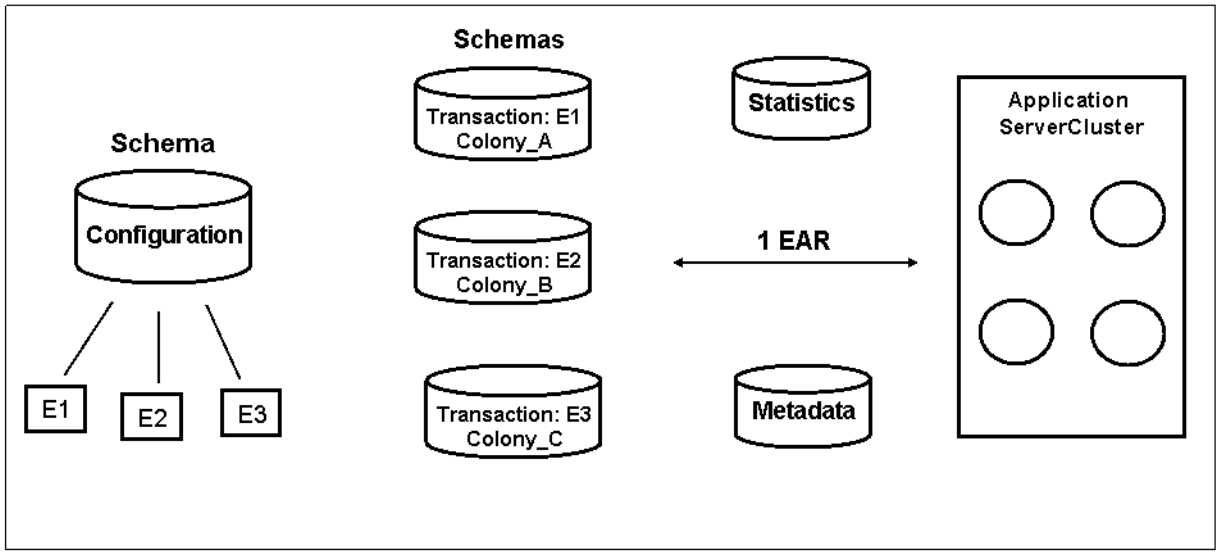
Single-Instance Multischema Deployment

A multischema deployment is another version of a single instance deployment, but with multiple database schemas. Enterprises can share the same Configuration schema but have different Transaction schemas, one for each set of enterprises. Note that each enterprise-specific Transaction schema is further associated with a Colony ID.

Note: For more detailed information about colonies, multischema tables, and how multischema features are implemented by Selling and Fulfillment Foundation, see Multischema Overview.

The following multischema deployment example shows three colonies:

- Colony_A = Configuration E1 + Transaction E1 schemas
- Colony_B = Configuration E2 + Transaction E2 schemas
- Colony_C = Configuration E3 + Transaction E3 schemas



In this type of deployment, the Configuration schema is shared by all three enterprises, but each enterprise has its own Transaction schema.

Following are the advantages and limitations of this multischema deployment.

Advantages:

- Central point of control because it is a single-instance deployment.
- You can run one agent server that will process data across colonies.
- You do not need to synchronize Configuration data, such as hub-level pipelines, that is shared by two colonies.
- Colonies can upgrade independently. However, if an enterprise that was inheriting rules from another enterprise is upgrading, the Configuration schema of both enterprises needs to be upgraded.

If an enterprise upgrades, its participating organizations should also upgrade. These include nodes, inventory organization, capacity organization, catalog organization, and so on.

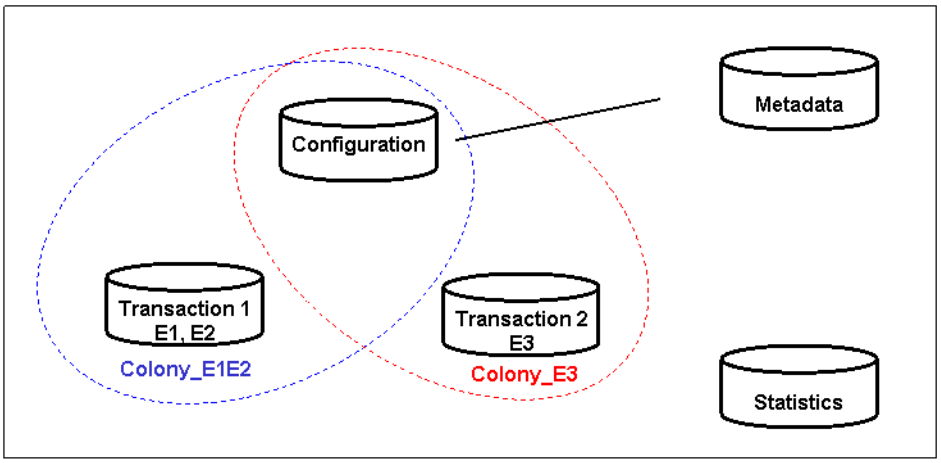
- Enterprises in different colonies can share the same Configuration schema, but because they are in different colonies, they can still be upgraded independently.
- Enterprises in different schemas can inherit rules from a common organization. For more information, see Using Template Organizations.
- Two different Transaction schemas can inherit some configuration rules.
- Compared to a single-instance deployment, enterprise growth can be handled by adding smaller-sized database nodes, rather than expanding monolithic databases. Compared to a multiple-instance deployment, the capacity of the application server is better utilized.
- Enterprises can benefit from the isolated Configuration schema when using the CDT, and at the same time, be able to deploy Configuration data across multiple colonies. For more information, see the *Selling and Fulfillment Foundation: Configuration Deployment Tool Guide*.

Limitations:

- Enterprises that belong to different colonies cannot share inventory, capacity, items, customers, and vendors.
- In the Configuration Deployment Tool (CDT), you can compare and migrate Configuration data all at once. But to migrate Master Data, you must point the CDT to each colony and migrate on a per-colony basis.
- Enterprises that belong to different colonies cannot participate with the same ship node. All participating organizations, including ship nodes, have to share the same Transaction schema.

Multischema Overview

Following is a simple diagram illustrating how multischema configurations work. In this diagram, two Transaction schemas, Transaction 1 and Transaction 2, share the same Configuration schema.



In this example, Transaction data for two of the enterprises, E1 and E2, resides in the Transaction 1 database, while data for enterprise E3 resides in the Transaction 2 database. You can create a colony to bound the set of enterprises that share the same Transaction schema.

Colonies

A colony is a set of database schemas required to provide the complete multischema functionality offered by Selling and Fulfillment Foundation. Every colony consists of the following schemas:

- Configuration schema - Contains Configuration data and is shared across colonies.
- Transaction schema - Contains Transaction data and is dedicated to a colony.
- Metadata schema - Contains database lookup information that directs transactions to the correct colony. This schema is shared across colonies and across versions.
- Statistics schema - Contains statistics for the application, such as statistics for APIs and agents, and is shared across schemas and versions.

Each colony has a Colony ID, such as Colony_E1E2. This colony name can be up to 40 characters in length. A unique, two-digit Colony Prefix further defines the colony. This can be from 10 – 99 (except 19 and 20). A DEFAULT Colony ID (Colony Prefix 20) is provided with every installation of Selling and Fulfillment Foundation.

In the previous Colony illustration, the Colony IDs are Colony_E1E2 and Colony_E3. You can define the Colony ID with the Manage Colony tool at installation time or later, as long as you ensure that no system activity, such as creating orders, takes place during the interval between installation and adding colonies. For more information about this tool, see Introduction to Managing Colonies.

In a multischema deployment, colonies also form a discrete unit that must be upgraded independently. The enterprises that belong to this colony must be migrated at the same time.

When you put enterprises into separate colonies, they cannot share inventory organizations, catalog organizations, capacity organizations, customer organizations and master organizations; they cannot participate with each other. In addition, no API returns data across multiple colonies, including list APIs, such as getOrderList and getShipmentList.

Each of the above schemas contains groups of tables that are used in Selling and Fulfillment Foundation.

Multischema Entity Tables

Multischema deployments are achieved through the use of the following tables:

- Configuration Schema Tables - These groups of tables generally store system or business rules such as sourcing rules, routing guide, shipping preferences, and so on. In addition, these tables store organizations.

  Only one Configuration schema can be used per version and it must be used by all enterprises in the deployment that are on that version.
- Transaction Schema Tables - These groups of tables contain a high volume of constantly-changing data.

  In the Sterling Warehouse Management System and in Selling and Fulfillment Foundation, this data is relevant to orders and shipment tasks. It also includes inventory, inventory location, and capacity tables.

- Master Data Tables - These groups of tables reside along with Transaction tables. They contain data that is created through batch feeds and is often referenced by other organizations.

  In the Sterling Warehouse Management System, Warehouse Layout Tables are all marked as Master Data. These tables contain data relevant to location such as location size, capacity, storage code, and zone. They also contain zone enterprise dedication and SKU dedication.

  In Selling and Fulfillment Foundation, Master Data tables contain data relevant to catalogs, pricing, and customers.
  Transaction and Master Data tables must be physically located in the same schema because Selling and Fulfillment Foundation joins data across Master Data and Transaction tables.
- Metadata Tables - These groups of tables contain data required for database lookup of core configuration information. It is a set of unique tables across versions and performs “traffic direction” of transactions across multischema, multi-version deployments.
- Statistics Tables - These groups of tables are used for maintaining statistics for the application such as agent and API statistics. For more information about statistics monitoring, see the *Selling and Fulfillment Foundation: Installation Guide*.

  Only one Statistics schema can be used per version.

Facts

A Fact is an attribute that is used by an API or an agent in Selling and Fulfillment Foundation to identify which colony to connect to and retrieve data from. For example, if information is required about a specific order, Facts enable the Metadata schema to be used as a lookup to determine which Transaction schema to connect to.

Following are the types of Facts in Selling and Fulfillment Foundation:

- ColonyId - Identifies the colony based on the ColonyId.
- PrimaryKey - The first two digits of the Primary Key are used to identify a colony through the Colony Prefix. In other words, when you create a new order, all the Primary Keys of that order will be generated with the prefix of that colony.
- LoginId - Identifies the colony from the Metadata lookup table PLT_COLONY_MAP. This table maps each user's LoginId to the colony that the user belongs to.
- OrganizationCode - Identifies the colony from the Metadata lookup table PLT_COLONY_MAP. This table maps each organization to the colony it belongs to.

If none of the facts can identify a colony, the DEFAULT colony is used for accessing the data.

APIs

APIs identify Facts based on the attributes passed in the input. To know which attributes are Facts for an API, see the *Selling and Fulfillment Foundation: Javadocs* for that API. For example, createOrder's Javadocs state that EnterpriseCode is a Fact. So if createOrder is called with EnterpriseCode = "E1", the order will be created in E1's Colony Transaction schema.

Facts that are passed in the input of an API should all point to the same colony; that is, users must remain in the same colony's context during an API call.

Certain attributes can be passed in any API's input, even if the Javadocs do not categorize them as a Fact. In the case of ColonyID, Selling and Fulfillment Foundation does not pass this attribute in APIs, but it is included in the following table because you can optionally implement the application to pass ColonyId. All of these attributes should be present in the header element:

Fact Attribute	Fact Type
ColonyId	ColonyId
CallingOrganizationCode	OrganizationCode
EnterpriseCode	OrganizationCode
OrganizationCode	OrganizationCode

APIs can also work based on Primary Keys. For example, a user can call the scheduleOrder API with just the OrderHeaderKey in the input. The first two digits of the Primary Key help the API identify which Colony it should act on.

Note: The following APIs are called from the Application Consoles and fetch data only from the user's colony:

- getIntegrationErrorGroupDetails
- getIntegrationErrorGroupList
- getIntegrationErrorList
- reprocessAllIntegrationErrorsInGroup

Agents

For an agent, you can set the ColonyId attribute in the agent criteria. If ColonyId is not set, the agent works on the DEFAULT Colony. Some agents like Person Info Purge, Person Info Archive Purge, and Audit Purge also need a TableType attribute. This is because YFS_Audit and YFS_Person_Info tables exist with each schema and store data corresponding to their own respective schemas. If the TableType attribute is not set, the agent works for the default TableType for that table. For YFS_Person_Info, the default TableType is TRANSACTION. For YFS_Audit, the default TableType is CONFIGURATION.

For example, to purge Person Info records for a Transaction schema for a Colony_E3, set the ColonyId to Colony_E3, and set the TableType attribute to TRANSACTION. (In Release 9.0, a TableType of TRANSACTION and MASTER is the same because they both reside within the Transaction schema.)

See “A Time-Triggered Transaction Reference” in the *Selling and Fulfillment Foundation: Application Platform Configuration Guide* for more information about the TableType parameter for Person Info Purge and Person Info History Purge.

Users and Organizations

Users and organizations are treated as Master Data and reside in the Transaction schema. When you create users and organizations, corresponding records are created in a Metadata PLT_COLONY_MAP lookup table. When a user logs in, if the login API does not have an enterprise to which that user belongs, the PLT_COLONY_MAP lookup table determines which colony the user belongs to.

Migrate Data in a Multischema Deployment

During incremental configurations of Selling and Fulfillment Foundation, the Configuration Deployment Tool (CDT) migrates configuration data. The CDT contains multischema options for migrating colonies of enterprises.

Because multischema enterprises can have independent deployments, some enterprises may want to move to a higher version of Selling and Fulfillment Foundation sooner than the others. The CDT migrates organization-specific data based on Colony ID attributes.

If you are migrating data in a multischema deployment, Sterling recommends following this sequence of tasks:

1. Move HUB and Organization Driven data first.
2. If an enterprise inherits any rules, pipelines, or user exit implementations from another enterprise, move the parent enterprise or its DEFAULT colony first.
3. Move the child enterprise.

The *Selling and Fulfillment Foundation: Configuration Deployment Tool Guide* contains detailed instructions for comparing and migrating data in a multischema deployment.

Using Template Organizations

In a multischema environment, if an organization wants to inherit Configuration data from another organization, the best practise of creating a third, template organization eases future data migration.

For example, if Organization E1 in Colony 1 wants to inherit Configuration data from Organization E2 in Colony 2, introduce a template organization E3 in the Default colony. Have E1 and E2 inherit from E3. If you then want to upgrade or migrate data in E2, you simply have to move E1/Colony 1 and E3/Default Colony, with no impact on E2/Colony 2.

Sterling Analytics Reporting in a Multischema Deployment

Selling and Fulfillment Foundation Sterling Analytics, which provides query, analysis, and report capability, offers data sources based on schema type: Configuration, Metadata, Master Data, and Transaction. You can optionally create multischema data sources for Transaction and Master Data schema types and associate them with a Colony Prefix. Reports are done on a per-colony basis.

For more information, see the *Selling and Fulfillment Foundation: Business Intelligence Guide*.

Deployment Benefit Matrix

The following table summarizes the advantages of various types of deployment:

Feature	Single Instance Benefit	Multiple Instance Benefit	Multiple Context Root Benefit	Multi-Schema Benefit
Enterprises can share inventory/capacity/catalog.	X			
Ship nodes can be shared across enterprises. Organizations can participate with each other.	X			
In the Configuration Deployment Tool (CDT), Configuration data can be compared and migrated all at once.				
Enterprises can inherit rules from another enterprise within this same Configuration schema.	X			X
Each enterprise can have its own code extensions, events, templates, extensions to user interfaces, menus, and user exit implementations. If two enterprises have different inventory systems, they need to have different interfaces, too. These can all be defined by enterprise.	X	X	X	X
Each enterprise is configured separately and can have its own sourcing rules, pipelines, tasks, and so on. This isolation of configuration means that you can deploy and move data by enterprise in the Configuration Deployment Tool (CDT).	X	X	X	X
All enterprises are under a single point of control. For example, you can start and stop agents and monitor processes from one place.	X			X
Enterprises upgrade independently by enterprise, due to isolation of Transaction data.		X	X	X
Enterprises can increase database size and keep adding instances.		X		
Application server monitoring/maintenance can be performed at an aggregate level across all deployments.	X		X	X
Isolation of context roots can result in better and more dynamic utilization of hardware capacity, especially if you ensure that the enterprises sharing an application server require different peak usage periods.	X		X	X
Gain in memory utilization. For example, because Configuration data is not shared, configuration of resource permissions would be duplicated, resulting in approximately twice the memory consumption.	X			X

Feature	Single Instance Benefit	Multiple Instance Benefit	Multiple Context Root Benefit	Multi-Schema Benefit
You do not need to synchronize Configuration data, such as hub-level pipelines, that are shared by two colonies.	X			X
Compared to a single-instance deployment, enterprise growth can be handled by adding smaller-sized database nodes, rather than expanding monolithic databases. Compared to a multiple-instance deployment, the capacity of the application server is better utilized.				X
Enterprises can benefit from the isolated Configuration schema when using the Configuration Deployment Tool, and at the same time, be able to deploy Configuration across multiple colonies.				X

Move from Single-Schema Mode to Multischema Mode

When installing Selling and Fulfillment Foundation, the GUI and text-based installation processes prompt for whether you are installing a multischema deployment and, depending on your responses, set up multischema mode for you. When installing by a silent installation file, you must specify the multischema properties necessary if you want a multischema deployment, as described in the *Selling and Fulfillment Foundation: Installation Guide*.

If you are upgrading to Release 9.0, both single-schema and multischema upgrade modes are supported. You must be running a multischema deployment before you can upgrade in multischema mode. For more information about upgrading in single-schema and multischema modes, refer to the *Selling and Fulfillment Foundation: Upgrade Guide*.

If you are running a single-schema deployment and want to move to multischema deployment, follow these steps:

1. In the `sandbox.cfg` file, set the following properties:

   ```
   multischema.enabled=true
   multischema.version=8.5
   ```

2. After editing `sandbox.cfg`, run the setupfiles script so that the runtime property files are re-created with the updated values:

   (For UNIX or Linux) `<INSTALL_DIR>/bin/setupfiles.sh`

   (For Windows) `<INSTALL_DIR>/bin/setupfiles.cmd`

   When you perform the previous two steps, the Metadata, Configuration, Transaction, and Statistics schemas are automatically set up for you.

3. Run the Colony Map Synchronizer agent, as described in the "Time-Triggered Transaction Reference" appendix in the *Sterling Distributed Order Management: Configuration Guide*. This agent inserts or updates colony mappings of organizations and users in the PLT_COLONY_MAP table.

Note: When you run the agent for the first time, it populates the PLT_COLONY_MAP table. Initially, you must run the agent on the DEFAULT colony that is part of the Selling and Fulfillment Foundation installation. After completing these steps with the DEFAULT colony, you can complete them on other colonies.

4. Rebuild and deploy the EAR, as described in the *Selling and Fulfillment Foundation: Installation Guide*.

5. Once you have completed the previous steps, you can add colonies using the ManageColony tool, as described in Introduction to Managing Colonies.

Add an Enterprise to a Colony

After completing the steps and adding colonies as described in Move from Single-Schema Mode to Multischema Mode, you can add Enterprises to colonies as follows:

1. On the Installation Rule screen (Applications Manager> Application Platform > System Administration > Installation Rules) set the Inventory Consolidation Level, Capacity Consolidation Level, and Catalog Model to “Enterprise.” Do not set them to “Hub” (they are mutually exclusive).
2. On the Create Organization screen (Applications Manager> Application Platform > Participant Modeling > Participant Setup > Organization Search > Create New) specify whether the Organization is an Enterprise and enter the Assigned Colony Id for that Organization.

   If the Organization already exists as an Enterprise, simply enter the Assigned Colony Id for that Organization. The *Selling and Fulfillment Foundation: Application Platform Configuration Guide* explains this process.

Introduction to Managing Colonies

A default colony is always created when you install the Selling and Fulfillment Foundation. You can use the Manage Colony tool to create additional colonies. All colonies share the Configuration schema, Statistics schema, and Metadata schema, but they have separate Transaction/Masterdata schemas.

You can add colonies to your multischema configuration in the following ways:

- Add a Colony Using the GUI Wizard in Windows, UNIX, or Linux
- Add a Colony from Command-Line Interface (Windows, Linux, or UNIX)
- Add a Colony in Silent Install Mode

For more information about multischema deployments and colonies, see Introduction to Multitenant Configurations.

Note: To add colonies, your application must be in multischema mode. If you have installed the Selling and Fulfillment Foundation in single-schema mode or upgraded to Release 9.0 from a previous version, move to multischema mode by performing the steps described in Move from Single-Schema Mode to Multischema Mode.

Add a Colony Using the GUI Wizard in Windows, UNIX, or Linux

You can add colonies to your Selling and Fulfillment Foundation deployment as follows:

1. Run the Manage Colony Wizard by entering the following command:

   Windows: `<INSTALL_DIR>\bin manageColonyWizard.cmd`
   Linux or UNIX: `<INSTALL_DIR>/bin ./manageColonyWizard.sh`
2. The initial screeen dialog box is displayed.
3. At the Manage Colony screen, click **Next** to start the installation program.
4. At the Manage Colony screen, enter the Colony ID. This is the name of the colony you wish to create and it can be up to 40 characters in length. In the next field, enter the Colony Prefix, which is a two-digit colony prefix for the colony. This can be from 10 – 99 (except 19 and 20). Click **Next**.
5. Configure your Transaction/Master schema by entering the following information and click **Next**:
   - Database user name
   - Database password
   - Confirm database password
   - Database catalog name (for more information see the *Selling and Fulfillment Foundation: Installation Guide*)
   - Database host name (or IP address)
   - Database port

6. After you enter the database information and click **Next**, the Confirm Database Information screen for Transaction/Master displays the database account information you entered on the previous screen. This screen is read-only. If the information is correct, click **Next**. If any information needs to be changed, click **Back** to return to the previous screen and make changes.
7. If you have installed Sterling Analytics, the Manage Colony Wizard prompts for the following information. Otherwise, proceed to Step 8.
    a. Configure your Sterling Analytics Transaction/Master schema by entering the following information and click **Next**.
        - Database user name
        - Database password
        - Confirm database password
        - Database catalog name (for more information see the *Selling and Fulfillment Foundation: Installation Guide*)
        - Database host name (or IP address)
        - Database port
    b. After you enter the database information and click **Next**, the Confirm Sterling Analytics Database Information Transaction/Master Schema screen displays the database account information you entered on the previous screen. This screen is read-only. If the information is correct, click **Next**. If any information needs to be changed, click **Back** to return to the previous screen and make changes.
    c. Configure your Sterling Analytics Staging schema by entering database information and click **Next**.
    d. After you enter the database information and click **Next**, the Confirm Database Account Information for Sterling Analytics Staging Schema screen displays the database account information you entered on the previous screen. This screen is read-only. If the information is correct, click **Next**. If any information needs to be changed, click **Back** to return to the previous screen and make changes.
8. When the installation is finished, the message *installation completed* is displayed.
9. If you wish to add another colony to your deployment, go back to Step 1 and repeat the procedure again, entering database information for the next colony.

This GUI installation writes an `<INSTALL_DIR>/addColony.xml` file, which contains the colony configuration data you entered during this procedure.

Add a Colony from Command-Line Interface (Windows, Linux, or UNIX)

You can add colonies to your Selling and Fulfillment Foundation deployment as follows:

1. Run the Manage Colony installation by entering the following command from the <INSTALL_DIR>/bin directory:

   Windows: `<INSTALL_DIR>\bin manageColonyWizard.cmd`
   UNIX or Linux: `<INSTALL_DIR>/bin ./manageColonyWizard.sh`

2. At the *Enter Colony ID* prompt, enter the Colony ID, which is the name of the colony you wish to create.
3. At the next prompt, enter the Colony Prefix, enter a two-digit colony prefix for the colony. This can be from 10 – 99 (except 19 and 20). Press **Enter**.
4. The installation guides you through entering database account information for Transaction/Master Schema. You are prompted separately for each of the following items. Enter a value for the first item, then press **Enter**. The second item will be displayed. Enter a value and press **Enter**. Repeat for each item:
   - Database user name
   - Database password
   - Confirm database password
   - Database catalog name (for more information see the *Selling and Fulfillment Foundation: Installation Guide*
   - Database host name (or IP address)
   - Database port
5. After you enter the database information and press **Enter**, the Confirm Database Information screen (read-only) for Transaction/Master displays the database account information you entered. Review each item's value, and press **Enter** to accept each one.
6. The installation program verifies the database connection. If a connection cannot be established, you receive an error and can re-enter the database information to make another connection attempt. If you cannot make a connection, consult with your database administrator.
7. If you have installed Sterling Analytics, the installation prompts for the following information. Otherwise, proceed to Step 8.
   a. The installation guides you through entering database account information for the Sterling Analytics Transaction/Master schema. You are prompted separately for each of the following items. Enter a value for the first item, then press **Enter**. The second item will be displayed. Enter a value and press **Enter**. Repeat for each item:
      - Database user name
      - Database password
      - Confirm database password

- Database catalog name (for more information see the *Selling and Fulfillment Foundation: Installation Guide*)
- Database host name (or IP address)
- Database port

b. After you enter the database information and press **Enter**, the Confirm Sterling Analytics Database Information Transaction/Master Schema screen (read-only) displays the database account information you entered. Review each item's value, and press **Enter** to accept each one.

c. The installation program verifies the database connection. If a connection cannot be established, you receive an error and can re-enter the database information to make another connection attempt. If you cannot make a connection, consult with your database administrator.

d. The installation guides you through entering database account information for the Sterling Analytics Staging Schema. You are prompted separately for each database item. Enter a value for each item and press **Enter**.

e. After you enter the database information and press **Enter**, the Confirm Database Account Information for Sterling Analytics Staging schema screen (read-only) displays the database account information you entered. Review each item's value, and press **Enter** to accept each one.

8. When the installation is finished, the system displays the message *installation completed*.
9. If you wish to add another colony to your deployment, go back to Step 1 and run the manageColonyWizard script again, entering database information for the next colony.

This command-line installation writes an `addColony.xml` file, which contains the colony configuration data you entered during this procedure.

Add a Colony in Silent Install Mode

You can create colonies in silent install mode by creating an `addColony.xml` file that you can pass with the `manageColony` command. This file contains colony and database information that is required when you add a colony. In addition, you can specify multiple passwords and their effective dates, predefining passwords for a given pool months in advance. These passwords will change on the fly without a server restart. Adding colonies in silent install mode automates the process and limits manual interaction.

You can add colonies to your Selling and Fulfillment Foundation deployment as follows:

1. Create a silent `addColony.xml` file with a text editor. The file must contain the entries described in the following table. This table is followed by a sample `addColony.xml` file. After you edit the file, make note of its name and location, so that you can pass it with the `manageColony` command.
2. Enter the following command to add a colony:

   Windows: `<INSTALL_DIR>\bin manageColony.cmd addColony.xml`
   UNIX or Linux: `<INSTALL_DIR>/bin ./manageColony.sh addColony.xml`

Repeat this command as necessary each time you want to create a colony, passing the XML file that is associated with each colony.

The following table describes the parameters in the `addColony.xml` file:

Parameter	Definition
<*colony_name*>	The name of the colony you want to create. This can be up to 40 characters in length.
<*primary_key_prefix*>	A two-digit prefix for the colony. This can be any number from 10 – 99 (except 19 and 20).
<*new_pool_id*>	The new connection pool pointing to the schema this colony is associated with.

<*jdbc_url*>	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 ServerHostname>:<PortNumber>;DatabaseName=<Database name>`
	◆ If using DB2, set to:
	`jdbc:db2://<Database ServerHostname>:<Port Number>/<Database name>.<db_user> Database user name`

<*db_user*>	Specify the user name associated with the database.
<*db_password*>	Specify the password associated with the database.

<*db_driver_class*>	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`

<*db_schema*>	Specify the schema name associated with the database if it is different from the <*db_user*> name you entered. **Note:** This parameter is case-sensitive and you must specify it in UPPERCASE.
<password.1>	Specify a database password for the effectivity date parameter specified in <effective.1>
<password.2>	Specify a database password for the effectivity date parameter specified in <effective.2>
<effective.1>	Specify an effective date for the <password.1> parameter shown above.
<effective.2>	Specify an effective date for the <password.2> parameter shown above.

Sample Silent AddColony File

The following sample file contains the parameters described in Add a Colony in Silent Install Mode. If you want to add more than one colony, create additional `addColony.xml` files that specify the colony and database information associated with those colonies.

Notes:

- The following example shows a schema and corresponding pool parameters for “BI” or Sterling Analytics, in case you are installing it.
- You can change only the properties displayed in italics in the sample `addColony.xml` file.

Sample addColony.xml

```
<colonyconfig>
    <colonies>
        <colony name="<colony_name>" pkprefix="<primary_key_prefix>" version="9.0">
            <schema poolid="DEFAULT_METADATA" tabletype="METADATA"/>
            <schema poolid="DEFAULT_CONFIGURATION_90" tabletype="CONFIGURATION"/>
            <schema poolid="DEFAULT_STATISTICS_90" tabletype="STATISTICS"/>
            <schema poolid="<new_pool_id_1>" tabletype="TRANSACTION"/>
            <schema poolid="<new_pool_id_1>" tabletype="MASTER"/>
            <schema poolid="<new_pool_id_2>" tabletype="BI"/>
            <schema poolid="<new_pool_id_3>" tabletype="BI_STAGING"/>
        </colony>
    </colonies>
    <pools>
        <pool id="<new_pool_id_1>">
            <jdbc>
                <param name="url" value="<jdbc_url>"/>
                <param name="user" value="<db_user>"/>
                <param name="password" value="<db_password>"/>
                <param name="driver" value="<db_driver_class>"/>
                <param name="schema" value="<db_schema>"/>
                <param name="password.1" value="<password_1>"/>
                <param name="password.2" value="<password_2>"/>
                <param name="effective.1" value="<effective_1>"/>
                <param name="effective.2" value="<effective_2>"/>
            </jdbc>
        </pool>
        <pool id="<new_pool_id_2>">
            <jdbc>
                <param name="url" value="<jdbc_url>"/>
                <param name="user" value="<db_user>"/>
                <param name="password" value="<db_password>"/>
                <param name="driver" value="<db_driver_class>"/>
                <param name="schema" value="<db_schema>"/>
                <param name="password.1" value="<password_1>"/>
                <param name="password.2" value="<password_2>"/>
                <param name="effective.1" value="<effective_1>"/>
                <param name="effective.2" value="<effective_2>"/>
            </jdbc>
        </pool>
        <pool id="<new_pool_id_3>">
            <jdbc>
                <param name="url" value="<jdbc_url>"/>
                <param name="user" value="<db_user>"/>
                <param name="password" value="<db_password>"/>
                <param name="driver" value="<db_driver_class>"/>
                <param name="schema" value="<db_schema>"/>
                <param name="password.1" value="<password_1>"/>
                <param name="password.2" value="<password_2>"/>
                <param name="effective.1" value="<effective_1>"/>
                <param name="effective.2" value="<effective_2>"/>
            </jdbc>
        </pool>
    </pools>
</colonyconfig>
```

Index

A

add colony
  silent mode, 25

addColony.xml, 25
  sample silent file, 27

AddingColoniesSilent, 25

agents, 16

analytics reporting, 17

APIs, 15

C

colonies
  colony ID, 13
  default, 13
  definition, 13
  example, 13

Colony, 27

colony prefix, 13

configuration schema tables, 14

D

default colony, 13, 16

deployments
  multiple context-root, 9
  multiple-instance, 8
  single-instance multischema, 11
  single-instance single schema, 6

E

entity tables, 14

F

facts, 15
  attributes and types, 15

File, 27

H

HUB data, 16

L

lookup table PLT_COLONY_MAP, 15, 20

M

manage colony tool, 22

manage colony wizard
  command-line interface, 24
  GUI, 22

manageColony command, 25

manageColony command parameters, 26

master data, 16

master data tables, 14

metadata tables, 14

migrating data, 16
  sequence, 16
  template organizations, 17

multiple context-root deployment
  advantages, 9
  limitations, 9

multiple-instance deployment
  advantages, 8
  limitations, 8

multischema deployment
  advantages, 11
  agents, 16
  analytics reports, 17
  APIs, 15
  colonies, 13
  entity tables, 14
  facts, 15
  limitations, 12

migrating data, 16
primary key, 15
multitenant
definition, 5

O

organization driven data, 16

P

PLT_COLONY_MAP lookup table, 15, 16
primary key, 15

S

schema tables, 14
single-instance deployment
advantages, 6
limitations, 7
Single-Instance Multischema Deployment, 5
statistics tables, 14

T

template organizations, 17
third-party logistics, 5
transaction schema tables, 14

Z

3PL businesses, 5