HP SOA Systinet

Software Version: 3.00

Installation and Deployment Guide

Document Release Date: June 2008
Software Release Date: June 2008

Legal Notices

Warranty

The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

The information contained herein is subject to change without notice.

Restricted Rights Legend

Confidential computer software. Valid license from HP required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Third-Party Web Sites

HP provides links to external third-party Web sites to help you find supplemental information. Site content and availability may change without notice. HP makes no representations or warranties whatsoever as to site content or availability.

Copyright Notices

Trademark Notices

Java™ is a US trademark of Sun Microsystems, Inc. Microsoft®, Windows® and Windows XP® are U.S. registered trademarks of Microsoft Corporation. IBM®, AIX® and WebSphere® are trademarks or registered trademarks of International Business Machines Corporation in the United States and/or other countries. BEA® and WebLogic® are registered trademarks of BEA Systems, Inc.

Contents

About this Guide

Welcome to HP SOA Systinet, the foundation of Service Oriented Architecture, providing an enterprise with a single place to organize, understand, and manage information in its SOA. The standards-based architecture of SOA Systinet maximizes interoperability with other SOA products.

HP Software controls access to components of SOA Systinet with a license. This document describes the full functionality of SOA Systinet including licensed components. If your license does not include these licensed components, their features are not available.

How This Guide is Organized

This guide describes how to setup an environment and deploy SOA Systinet to it.

This guide contains the following chapters:

- Chapter 1, Getting Started

  Design your environment for SOA Systinet.

- Chapter 2, Basic Installation

  Use the GUI Installer, designed for test and evaluation installations of SOA Systinet.

- Chapter 3, Setting Up Production Environments

  Setup and configure a production environment for SOA Systinet.

- Chapter 4, Deploying SOA Systinet

  Deploy SOA Systinet to a production environment using command-line installation.

- Chapter 5, After Installation

Additional processes that may be required after deploying SOA Systinet.

Document Conventions

This document uses the following typographical conventions:

run.bat make	Script name or other executable command plus mandatory arguments.
[--help]	Command-line option.
either | or	Choice of arguments.
replace_value	Command-line argument that should be replaced with an actual value.
{arg1 | arg2}	Choice between two command-line arguments where one or the other is mandatory.
`rmdir /S /Q System32`	User input.
`C:\System.ini`	Filenames, directory names, paths and package names.
`a.append(b);`	Program source code.
`server.Version`	Inline Java class name.
`getVersion()`	Inline Java method name.
Shift+N	Combination of keystrokes.
Service View	Label, word, or phrase in a GUI window, often clickable.
OK	Button in a user interface.
New→Service	Menu option.

Documentation Updates

This guide's title page contains the following identifying information:

- Software version number, which indicates the software version.
- Document release date, which changes each time the document is updated.
- Software release date, which indicates the release date of this version of the software.

To check for recent updates, or to verify that you are using the most recent edition of a document, go to:

http://h20230.www2.hp.com/selfsolve/manuals

This site requires that you register for an HP Passport and sign-in. To register for an HP Passport ID, go to:

http://h20229.www2.hp.com/passport-registration.html

Or click the **New users - please register** link on the HP Passport login page.

You will also receive updated or new editions if you subscribe to the appropriate product support service. Contact your HP sales representative for details.

Support

You can visit the HP Software Support Web site at:

http://www.hp.com/go/hpsoftwaresupport

HP Software Support Online provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by using the HP Software Support web site to:

- Search for knowledge documents of interest
- Submit and track support cases and enhancement requests
- Download software patches
- Manage support contracts
- Look up HP support contacts
- Review information about available services
- Enter into discussions with other software customers
- Research and register for software training

Most of the support areas require that you register as an HP Passport user and sign in. Many also require a support contract.

To find more information about access levels, go to:

http://h20230.www2.hp.com/new_access_levels.jsp

To register for an HP Passport ID, go to:

http://h20229.www2.hp.com/passport-registration.html

1 Getting Started

Before installing SOA Systinet you must make sure that the environment you want to install to is appropriate and suitable for your needs.

The following sections describe the requirements and options available:

- Prerequisites on page 11
- Supported Platforms on page 12
- Designing Your Deployment on page 14

Prerequisites

The following hardware and software is required to run SOA Systinet:

Hardware

Hardware requirements vary depending on sizing and deployment type.

For a distributed, production environment, the requirements are:

- For each physical node, an Intel Pentium Dual Core processor, 2 GB RAM, 1 GB free disk space and a network card that supports 1 Gb/sec.
- Network bandwidth of 1 Gb/sec or higher.

For development and evaluation purposes, SOA Systinet can run on a single machine, even on a notebook.

The hardware requirements in this case are:

- Intel Pentium IV processor, 1 GB RAM, 1 - 2 GB free disk space and a network card that supports 100 Mb/sec.

- Network bandwidth of 100Mb/sec or higher.

Software

Each physical node must have the following software:

- A JDK and a J2EE application server from the list in Supported Platforms on page 12. The application server must use this JDK.
- A `JAVA_HOME` environment variable set to point to the Java JDK used by the host J2EE application server.
- Access to a supported database from Supported Platforms on page 12.

Supported Platforms

SOA Systinet supports the following browsers:

- Microsoft Internet Explorer 6 and 7
- Mozilla Firefox 2

SOA Systinet supports the following combinations of application servers, JDKs and backend databases.

The combinations are described in the following tables, organized by operating system:

Supported Platforms on Windows

Windows Server 2003 on x86 Platforms		
J2EE Application Server	**Java JDK**	**Relational Database**
BEA® WebLogic Server ® 9.2 MP2	Sun JDK 5.0_u9	MSSQL 2005
		Oracle 10g
IBM WebSphere® 6.1.0.13	IBM JDK 1.5	Oracle 10g
JBoss® 4.2.2	Sun JDK 5.0_u9	MSSQL 2005
		Oracle 10g
Oracle Application Server 10.1.3.3	Sun JDK 1.5.0_09 or higher	Oracle 10g

If you use JBoss, install it with a path less than 20 characters. This limitation is caused by JBoss expanding the application in the local disk and the Windows 255 character limit on path names.

Supported Platforms on Linux

RedHat Enterprise Linux 4.0 and 5.0 on x86 Platforms		
J2EE Application Server	**Java JDK**	**Relational Database**
BEA® WebLogic Server ® 10.0	JRocket	Oracle 10g
JBoss® 4.2.2	Sun JDK 5.0_u9	Oracle 10g
Oracle Application Server 10.1.3.3	Sun JDK 5.0_u9	Oracle 10g

Supported Platforms on AIX

AIX 5L 5.3 on PowerPC Platforms		
J2EE Application Server	**Java JDK**	**Relational Database**
IBM WebSphere® 6.0.1.13	IBM JDK 1.5	DB2 9.1

Supported Platforms on HP-UX

HP-UX 11.23 on Itanium Platforms		
J2EE Application Server	**Java JDK**	**Relational Database**
BEA® WebLogic Server ® 9.2 MP2	HP JDK 1.5	Oracle 10g
Oracle Application Server 10.1.3.3	HP JDK 1.5	Oracle 10g

HP-UX 11.31 on Itanium Platforms		
J2EE Application Server	**Java JDK**	**Relational Database**
BEA® WebLogic Server ® 9.2 MP2	HP JDK 1.5	Oracle 10g

Designing Your Deployment

HP SOA Systinet can be deployed on a wide range of scales. You must design your deployment to match the scale of your network and your own J2EE application installation procedures. Broadly speaking, there are the following types of deployment:

- Development on page 14

  To evaluate the product, you can deploy it on a single machine.

- Production on page 15

  To use SOA Systinet in a production environment, cluster it with SOA Systinet installed on multiple nodes.

- Upgrade on page 16

  Upgrading from and earlier version of SOA Systinet.

Development

If you are a developer, CIO, or other IT manager who wants to learn the functions of SOA Systinet, this is the correct type of deployment for you. It should be on one machine and preferably on one J2EE server instance. The simplest approach is to deploy SOA Systinet to the JBoss application server.

Use the installation wizard to deploy the product to JBoss, following the default settings. Server configuration for JBoss is handled within this wizard and in the `serverstart` and `serverstop` scripts.

If you use an application server other than JBoss, the installation wizard can only create EAR files, which you then deploy using the application server tools. You must also modify server classpaths, configure JMS and set Java properties yourself.

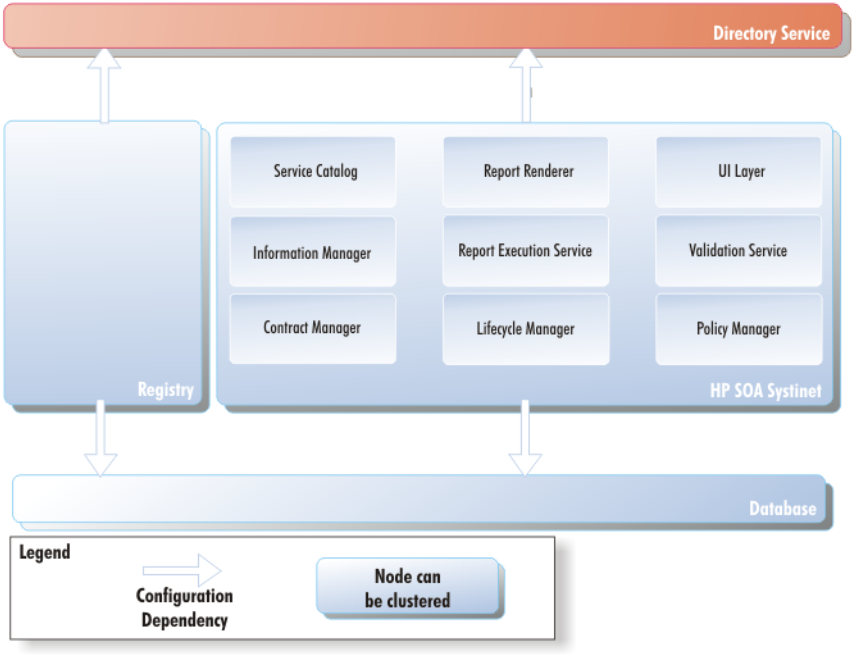
For installation details, see Chapter 2, Basic Installation.

Production

Deploying SOA Systinet for use in a production environment is complex. SOA Systinet is likely to be clustered and linked to a database and directory service on separate machines. A schematic of a production deployment is shown in Figure 1. If you are creating such a deployment, you should already have a set of tools and procedures for deploying J2EE applications and managing relational databases.

For details, see Chapter 3, Setting Up Production Environments.

Figure 1. Production Environment Deployment



When you deploy SOA Systinet to a production environment you may need additional configuration options that are not available in the GUI installer.

You can install SOA Systinet using the command line.

For details, see Chapter 4, Deploying SOA Systinet.

Upgrade

If you have an installation of SOA Systinet 2.52, you can upgrade to SOA Systinet 3.00.

To upgrade from SOA Systinet 2.52

1 Install SOA Systinet 3.00 according to your deployment requirements.

2 If you have customized extensions in SOA Systinet 2.52, apply them to SOA Systinet 3.00:

   a Install HP SOA Systinet Customization Editor as part of HP SOA Systinet Workbench 3.00.

     For details, see the *HP SOA Systinet Customization Editor Guide*.

   b Open the 2.52 extensions in Customization Editor 3.00.

     For details, see the *HP SOA Systinet Customization Editor Guide*.

   c Build the extensions in Customization Editor 3.00.

     For details, see the *HP SOA Systinet Customization Editor Guide*.

   d Apply the extensions to SOA Systinet 3.00.

     For details, see "Applying Extensions" in the *HP SOA Systinet Administration Guide* .

3 Migrate your data from SOA Systinet 2.52 to SOA Systinet 3.00.

   For details, see "Data Migration" in the *HP SOA Systinet Administration Guide* .

2 Basic Installation

The GUI installer is the easiest way to install SOA Systinet. However, it may not be suitable for all the configuration options required by production environments.

Production installation and deployment is described in detail in Chapter 4, Deploying SOA Systinet.

If you want to use the GUI installer for a production environment or for a non-JBoss application server, you should read Chapter 3, Setting Up Production Environments before running the installation.

This chapter contains the following sections:

Using the GUI Installer

Before using the GUI Installer make sure that you have a correctly setup environment.

For hardware and software requirements, and supported platforms, see Chapter 1, Getting Started.

For an evaluation environment, you need valid credentials to a configured database. For details, see Preparing Databases on page 35.

JBoss does not require any additional configuration for evaluation purposes. If you are using the GUI installation for a production environment with JBoss or for a different application server, see Setting Up Application Servers on page 46.

To install using the Installation Wizard:

1. Make sure the application server is not running.
2. Do one of the following:

- Execute the file `hp-soa-systinet-3.00.jar`, located on the installation CD or in your distribution directory.
- Execute the following command:

  java -jar hp-soa-systinet-3.00.jar

  Additional command line options are available.

  For details, see Installation Command Line Options on page 112.

The Installation Wizard opens displaying the Welcome page.

3 In the Welcome page, review the hardware and software requirements, and then click **Next**.

The License page opens.

4 In the License page. review the license, select **I Accept the Terms of the License Agreement**, and then click **Next**.

The Installation Folder page opens.

5 In the Installation Folder page, type or **Browse** for the location you want to use as your SOA Systinet installation folder, and then click **Next**.

The installer unpacks the distribution files to the chosen location.

The Application Server page opens.

The installation location is referred to as `SOA_HOME` throughout the documentation.

6 In the Application Server page, select the application server to use and click **Next**.

The Scenario Selection page opens.

For evaluation purposes, HP Software recommend JBoss. Other application servers require additional setup and configuration.

For details of production deployment, see Chapter 4, Deploying SOA Systinet.

7 In the Scenario Selection page, select **Default**, and then click **Next**.

The Password Encryption page opens.

The **Advanced** scenarios enable you to perform parts of the installation separately. These functions are duplicated by the Setup Tool and are discussed as administration functions.

For details, see the "Setup Tool" section in the *HP SOA Systinet Administration Guide* .

8 In the Password Encryption page, do one of the following:

- Select **Enable**, input the Master Passphrase and Confirm Passphrase, and then click **Next**.
- Select **Disable**, and then click **Next**.

The installer validates the encryption and the License Information page opens.

9 In the License Information page, do one of the following and then click **Next**:

- Select **Install a 60 day evaluation license**.
- Select **Enter license details** and input the license details provided by your sales representative.

The Updates page opens.

The administrator can change the license at a later date.

For details, see "Managing the License" in the *HP SOA Systinet Administration Guide* .

10 In the Updates page, use **Add** and **Remove** to select updates to apply during installation, and then click **Next**.

The Custom Extensions page opens.

11 In the Custom Extensions page, use **Add** and **Remove** to select extensions to apply during installation, and then click **Next**.

The Database Selection page opens.

12 In the Database Selection Page, select the database type to use, and then click **Next**.

The Database Setup Operations page opens.

13 In the Database Setup Operations page, do one of the following:

- Select **Create Database**, and click **Next**.
- Select **Create Schema**, and click **Next**.

A different Database Options page opens depending on the database and whether you are creating a database or a schema.

For more details about database setup options, see Database Installation Types on page 36.

14 In the Database Options page, input the parameters you want, and then click **Next**.

For parameter details, see the following sections:

- DB2 Create Database on page 25
- DB2 Create Schema on page 26
- MSSQL Database Parameters on page 27
- Oracle Create Database on page 28
- Oracle Create Schema on page 30

The JDBC Drivers page opens.

15 In the JDBC Drivers page, type or **Browse** for the driver to use, and then click **Next**.

For multiple drivers, use a comma to separate them.

The installer validates the database parameters, the configuration tables, and the driver.

The Endpoint Properties page opens.

16 In the Endpoint Properties page, specify the endpoint properties, and then click **Next**.

The User Management page opens.

If you change the port numbers from their default values, you must also change the application server configuration to use these ports.

For JBoss details, see Configuring JBoss Port Numbers on page 57.

17 In the User Management page do one of the following:

- Do not select **LDAP** and click **Next** to store accounts in your database and continue to Step 18.
- Select **LDAP** to integrate with an LDAP server account store.

  The installer continues with LDAP settings as described in LDAP Options on page 31.

18 In the Administrator Account page set the administrator credentials, and then click **Next**.

The Repository Import page opens.

19 In the Repository Import page, do one of the following:

- Select **Initial Import**, and then click **Next**.
- Select **Custom Import**, and type or **Browse** for a custom image, and then click **Next**.

The installer validates the data image.

For JBoss, the JBoss Deployment Properties page opens.

For other applications servers, skip to Step 21, the SMTP Server Authentication page.

20 In the JBoss Deployment Properties page, type or **Browse** for the JBoss installation folder and deployment directory, and then click **Next**.

The installer verifies the data source and the JBoss settings.

The SMTP Server Authentication page opens.

21 If you want mail notifications set the server and email.

To authenticate, select **Authenticate** and enter the SMTP server credentials.

Click **Next** to continue.

The installer creates the client package and the Confirmation page opens.

22 In the Confirmation page, click **Next** to start the installation process.

The Installation Progress page opens.

23 In the Installation Progress page, track each step of the installation. When the installation is complete, click **Next**.

The Installation Finished page opens.

24 Click **Finish** to exit the Installation Wizard.

You can review the installation and configuration settings in the log file:

```
SOA_HOME/logs/install.log
```

Depending on the installation options, you may need to perform some additional operations after installing SOA Systinet.

For details, see After GUI Installation on page 33.

Database Installation Parameters

The Database Options page varies depending on your installation settings.

For details, see the following sections:

DB2 Create Database

To create a new tablespace in DB2, set the following parameters:

Table 1. DB2 Create Tablespace Parameters

Parameter	Description	Notes
Database Server Address	The hostname or IP address where the database server is accessible.	In the database connection string `jdbc:db2:@dbhost42:50000:platform`, the hostname is `dbhost42`.
Database Server Port	The connection port for the database.	In the database connection string `jdbc:db2:@dbhost42:50000:platform`, the port number is `50000`.
Existing Database Name	The name of the database.	In the database connection string `jdbc:db2:@dbhost42:50000:platform`, the database name is `platform`.
Database Administrator Name	The user name and password of the administrator of the database.	
Database Administrator Password		
New Database Tablespace	Name of the tablespace you create with the **Create Database** option.	You must type a new tablespace name when you create a database. When you type a new tablespace name, the tablespace name must not conflict with existing objects in the database.
Tablespace Datafile	The path to the tablespace datafile that is stored on the database host machine.	
Existing Database User Name	The name and password of an existing database user.	
Database User Password		
Buffer Pool /with 16k page size/	The buffer pool to use for the tablespace.	

DB2 Create Schema

To create a new schema in DB2, set the following parameters:

Table 2. DB2 Create Schema Parameters

Parameter	Description	Notes
Database Server Address	The hostname or IP address where the database server is accessible.	For example, in the database connection string `jdbc:db2:@dbhost42:50000:platform`, the hostname is `dbhost42`.
Database Server Port	The connection port for the database.	For example, in the database connection string `jdbc:db2:@dbhost42:50000:platform`, the port number is `50000`.
Existing Database Name	The name of the database.	For example, in the database connection string `jdbc:db2:@dbhost42:50000:platform`, the database name is `platform`.
Existing Database User Name	The user name and password of an existing database user.	
Database User Password		
Database Tablespace	The tablespace to use for the new schema.	

MSSQL Database Parameters

To create a new tablespace or schema in MSSQL, set the following parameters:

Table 3. MSSQL Database Parameters

Parameter	Description	Notes
Database Server Address	The hostname or IP address where the database server is accessible.	For example, in the database connection string `jdbc:sqlserver:@sqlhost:1433:platform`, the hostname is `sqlhost`.
Database Server Port	The connection port for the database.	For example, in the database connection string `jdbc:sqlserver:@sqlhost:1433:platform`, the port number is `1433`.
New Database Name	The name of the database.	For example, in the database connection string `jdbc:sqlserver:@sqlhost:1433:platform`, the database name is `platform`.
Existing Database User Name	The user name and password of the user with tablespace or schema creation rights for the database.	
Database Administrator Password		

Oracle Create Database

To create a new tablespace in Oracle, set the following parameters:

Table 4. Oracle Create Database Parameters

Parameter	Description	Notes
Database Server Address	The hostname or IP address where the database server is accessible.	For example, in the database connection string `jdbc:oracle:thin:@orahost:1521:platform`, the hostname is `orahost`.
Database Server Port	The connection port for the database.	For example, in the database connection string `jdbc:oracle:thin:@orahost:1521:platform`, the port number is `1521`.
Existing Database Name	The name of the database.	For example, in the database connection string `jdbc:oracle:thin:@orahost:1521:platform`, the database name is `platform`.
Full Connection String	The full connection string to the database.	Select this as option as an alternative to inputting the individual connection parameters.
Database Administrator Name	The user name and password of the administrator of the database.	
Database Administrator Password		
New Database Tablespace	Name of the tablespace you create with the **Create Database** option.	You must type a new tablespace name when you create a database. When you type a new tablespace name, the tablespace name must not conflict with existing objects in the database.
Tablespace Datafile	The path to the tablespace datafile that is stored on the database host machine.	Only required to create a new database tablespace. Must not conflict with existing objects in the database.

Parameter	Description	Notes
New Database User Name	The name and password of a new database user.	When you create a new user, the user name must not conflict with existing objects in the database.
Database User Password		
Confirm Password		

Oracle Create Schema

To create a new schema in Oracle, set the following parameters:

Table 5. Oracle Create Schema Parameters

Parameter	Description	Notes
Database Server Address	The hostname or IP address where the database server is accessible.	For example, in the database connection string `jdbc:oracle:thin:@orahost:1521:platform`, the hostname is `orahost`.
Database Server Port	The connection port for the database.	For example, in the database connection string `jdbc:oracle:thin:@orahost:1521:platform`, the port number is `1521`.
Existing Database Name	The name of the database.	For example, in the database connection string `jdbc:oracle:thin:@orahost:1521:platform`, the database name is `platform`.
Full Connection String	The full connection string to the database.	Select this as option an alternative to inputting the individual connection parameters.
Database Administrator Name	The user name and password of the administrator of the database.	
Database Administrator Password		

LDAP Options

If you select **LDAP** in Step 17 of Using the GUI Installer on page 19, the installer continues with LDAP service settings.

For more details about LDAP integration, see LDAP Accounts Integration on page 108.

To set LDAP options:

1 In the LDAP Service page, set the LDAP connection parameters and credentials, and then click **Next**.

2 In the LDAP Search Rules page enter the parameters described in Table 6, and then click **Next**:

Table 6. LDAP Search Rules Properties

Property	Description	
Search Filter	The notation of the search filter conforms to the LDAP search notation. You can specify the LDAP node property that matches the user account or group.	
Search Base	LDAP will be searched from this base according to the **Search Scope** settings.	
Search Scope	One-level Scope	Only direct sub-nodes of the search base (entries one level below the search base) will be searched. The base entry is not included in the scope.
	Subtree Scope	The search base and all its sub-nodes will be searched.
Results Limit	Number of items returned when searching LDAP. If more results are returned by an LDAP search the remainder are disregarded and not shown.	

3 In the User Property Mapping page, use **Add** and **Remove** to set the property mappings, and then click **Next**.

The Group Properties page opens.

The following mandatory user account properties must be mapped from an LDAP server:

```
java.lang.String loginName
java.lang.String fullName
```

The following optional user account properties can be mapped from an LDAP server:

```
java.lang.String Email
java.lang.String Description
java.lang.String LanguageCode
java.lang.String Phone
java.lang.String AlternatePhone
java.lang.String Address
java.lang.String City
```

```
java.lang.String Country
java.lang.Boolean Blocked
```

4 In the Group Properties page enter the parameters described in Table 6, and then click **Next**.

The Group Property Mapping page opens.

5 In the Group Property Mapping page, use **Add** and **Remove** to set the property mappings, and then click **Next**.

This is the last LDAP options page and the installer continues with the Administrator Account Configuration page, Step 18.

The following mandatory group properties must be mapped from an LDAP server:

```
java.lang.String name
java.lang.String member
```

The following optional group properties can be mapped from an LDAP server:

```
java.lang.string Owner
java.lang.String Description
```

SOA Systinet logins are case sensitive by default. If want the login name to be case insensitive you must add the following property to `SOA_HOME/conf/setup/configuration.properties`:

```
shared.um.account.caseInsensitiveLoginName=true
```

You must also ensure that the application server uses matching case sensitive or insensitive authentication as well.

After GUI Installation

Depending on your deployment environment and intended use, there may be a number of steps required to complete your SOA Systinet deployment.

If you used the GUI installer with a non-JBoss application server, see Deploying the EAR File on page 128 for EAR deployment instructions.

If you used the GUI installer for a production deployment, see Chapter 5, After Installation for a list of procedures that may be required.

To start the SOA Systinet server, do one of the following:

- For JBoss execute the command:

  ```
  PLATFORM_HOME/bin/serverstart
  ```

- For other application servers, use the relevant server start procedure.

3 Setting Up Production Environments

Before installing SOA Systinet, your must prepare your environment.

Set up your environment as described in the following sections:

- Preparing Databases on page 35
- Setting Up Application Servers on page 46
- LDAP Accounts Integration on page 108

Preparing Databases

This section describes database administration tasks for SOA Systinet. The database administrator has tasks at the time of installation and may also have tasks when SOA Systinet is updated, extensions are applied, or data is migrated.

Before you can install SOA Systinet the database administrator must set up the database.

This chapter contains the following sections:

- Database Installation Types on page 36

  Explains the different database installation scenarios according to the required level of access to the database.

- Each database type requires specific prerequisites for SOA Systinet and the procedures for creating the user types determined by your installation scenario are different.

  For details, see the following sections:

  - Setting Up DB2 on page 38
  - Setting Up MSSQL on page 40

- Setting Up Oracle on page 43

Database Installation Types

There are several ways to arrange the database and schema for SOA Systinet.

The option required is usually determined by the level of access to the database.

The options are described in the following sections:

- Create Schema on page 36

  Connect to an existing database with privileges to create new tables in order to create a new schema for SOA Systinet data.

- Create Database on page 37

  Create a new database or tablespace with database administrator privileges to contain SOA Systinet data.

- Manual Database Arrangement on page 37

  Delegate database and table creation to the database administrator.

After installation, there are more database options available using the Setup Tool.

For details, see "Setup Tool" in the *HP SOA Systinet Administration Guide* .

Create Schema

The **Create Schema** option, available in the GUI installer and command-line deployment, creates tables and indexes in the default schema in an existing database/tablespace provided by the database administrator. Select this method if you have an account in a database with an empty schema (recommended) and privileges to create tables and indexes.

This documentation uses *power user* to describe a user with these privileges.

Create Database

The option to create a database is available in the GUI installer and command-line deployment. This option automates database arrangement as much as possible, but requires database administrator credentials. The process creates users, the database or tablespace depending on your database type, and continues with the creation of the schema.

- For DB2, this option requires an existing database, OS user and database administrator credentials.

  This option does not create a new physical database. It creates a tablespace in an existing database to separate repository data. The user is then granted privileges to use the tablespace, create tables, and connect to the database.

- For MSSQL, this option requires an existing user with the database creator role.

  This option creates a new physical database with collation inherited from the server settings.

- For Oracle, this option requires an existing database and database administrator credentials.

  This option does not create a new physical database. It creates a new tablespace to hold SOA Systinet data separately and creates a new database account which uses the new tablespace as its default tablespace.

Manual Database Arrangement

The database administrator may want to arrange the database manually.

- In some cases the database administrator (DBA) cannot share the DBA credentials required for the Create Database option or the *power user* credentials for the Create Schema option.

- The database administrator may want to amend the default DDL scripts. For example, to create indexes in a separate tablespace.

In these cases the database administrator must perform the database related installation operations manually as part of *Decoupled Database Installation*.

For details, see Decoupled Database Deployment on page 125.

Typically the database administrator creates a power user account for the SOA Systinet schema and a common user account with minimal privileges to insert, select, update, and delete SQL operations in power user tables.

The database administrator does not distribute the power user credentials and provides the common user credentials to the SOA Systinet administrator to configure the application server datasource.

Setting Up DB2

Configure the DB2 database as follows for use with SOA Systinet:

1. If you plan to use the SOA Systinet full text search feature, make sure the optional DB2 Net Search Extender is installed.
2. If one does not exist, create a database that uses the UTF-8 Code Set.
3. If it does not exist, create a bufferpool with a page size of 32k. Create a *system temporary tablespace* with 32k page size, using the bufferpool.
4. To ensure the successful import or export of large data images HP Software recommend increasing the log file size (`LOGFILSIZ`) parameter to 2048 or higher and the number of primary log files (`LOGPRIMARY`) parameter to 15 or higher.

   To ensure that there is sufficient memory HP Software recommend increasing the application heap size (`APPLHEAPSZ`) parameter to 512 or higher.
5. Create an OS user account to hold the SOA Systinet data.
6. Create accounts based on the database access required by SOA Systinet during the installation.

   For details, see Database Installation Types on page 36.

   - For the Create Database option no additional accounts are required.
   - For the Create Schema option, create a power user.

     For details, see Setting Up a DB2 Power User on page 39 .
   - For Manual Database Arrangement, create the database, create a power user account to own the schema, and create a common user account with minimal privileges.

     For details, see Setting Up a DB2 Power User on page 39 and Setting Up a DB2 Common User on page 39.

After installation, if you require SOA Systinet Full Text Search to be enabled, set up indexes on DB2.

For details, see Enabling Full Text Search on DB2 on page 147.

Setting Up a DB2 Power User

In order to use the Create Schema option during installation, the database administrator should create a *power user* with appropriate privileges to the database.

To set up a power user on DB2:

1. Grant CONNECT, CREATETAB, and IMPLICIT_SCHEMA privileges to the OS user account.
2. Grant use of the tablespace to the OS user account.

Setting Up a DB2 Common User

In cases where the database administrator restricts access to the database to just select, insert, update, and delete operations, SOA Systinet requires a user with these privileges.

The SOA Systinet schema must exist prior to creating the common user.

To set up a common user on DB2:

1. Create an OS account for the common user.
2. Grant the common user connection privileges.

   ```
   GRANT CONNECT ON DATABASE TO common_user
   ```

3. Open the DB2 Command Editor and connect to the database using *power user* credentials.
4. Grant the common user privileges to work with the schema.

```
SELECT 'GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE ' || TABNAME || '
       TO common_user;' FROM syscat.tables
       WHERE LOWER(tabschema) = LOWER('power_user')
```

5 After the results display, click **Fetch More Rows** at least twice until all rows are displayed.

6 Select all results and copy them to the clipboard.

7 In the **Commands** window, paste the clipboard contents.

8 Execute the commands.

9 Create aliases for the tables:

```
SELECT 'CREATE ALIAS ' || TABNAME || ' FOR power_user.'
     || TABNAME || ';' FROM syscat.tables
     WHERE LOWER(tabschema) = LOWER('power_user')
```

10 After the results display, click **Fetch More Rows** at least twice until all rows are displayed.

11 Select all results and copy them to the clipboard.

12 Open a new instance of DB2 Command Editor and connect to the database using the common user credentials.

13 In the common user **Commands** window, paste the clipboard contents.

14 Execute the commands.

Setting Up MSSQL

Use *SQL Server Configuration Manager* to configure the MSSQL database as follows for use with SOA Systinet:

1 Make sure that the TCP/IP protocol is enabled and set to a static port, for example 1433.

2 Running SOA Systinet requires XA transactions support (not required for installation).

For details about setting up XA transaction support, see http://msdn2.microsoft.com/en-us/library/aa342335.aspx .

3 If you want to use the full text search feature in SOA Systinet, make sure that the Full-Text Search engine is installed together with the database engine during the installation of MSSQL Server.

4 Create a login in the database server to hold SOA Systinet tables in the database. The login must have the *database creator* role.

The login must be able to access the master database for XA related stored procedures:

- Create a user in the master database for the login.
- Add the account to the SqlJDBCXAUser role.

5 Create users based on the database access required by SOA Systinet during the installation.

For details, see Database Installation Types on page 36.

- For the Create Database option the installer uses the login to automatically arrange the database.
- For the Create Schema option, if you want to separate the SOA Systinet data (recommended), use the login to create a database with given case sensitive collation.

You can create the database for another account, but you must then grant create table privileges to the new account.

The installer uses the login to create the schema in this new database.

The login must be able to access the master database with the appropriate role.

- For Manual Database Arrangement, use the power user login to create the database with given case sensitive collation, and create a common user account with minimal privileges.

  For details, Setting Up an MSSQL Common User on page 42.

  The login must be able to access the master database with the appropriate role.

After installation, if you require SOA Systinet Full Text Search to be enabled, set up indexes on MSSQL.

For details, see Enabling Full Text Search on MSSQL on page 148.

Setting Up an MSSQL Common User

In cases where the database administrator restricts access to the database to just select, insert, update, and delete operations, SOA Systinet requires a user with these privileges.

To set up a common user on MSSQL:

1 Open the MSSQL SQL Server Management Studio or the sqlcmd command line editor.

2 Create a common user login in the server and user in the database created for SOA Systinet (systinetdb).

The login must be able to access the master database for XA related stored procedures:

- Create a user in the master database for the login.
- Add the account to the SqlJDBCXAUser role.

3 Execute the following statements:

```
USE [master]
GO
CREATE LOGIN [common_user] WITH PASSWORD=N'...', DEFAULT_DATABASE=[master],
CHECK_EXPIRATION=OFF, CHECK_POLICY=OFF
GO
USE [systinetdb]
GO
CREATE USER [common_user] FOR LOGIN [common_user]
GO
```

4 Grant rights to the common user to read and write to SOA Systinet tables.

Execute the following statements:

```
USE [systinetdb]
GO
```

```
EXEC sp_addrolemember N'db_datawriter',N'common_user'
GO
USE [login_name]
GO
EXEC sp_addrolemember N'db_datareader', N'common_user'
GO
```

5 Add rights to the common user to use XA transactions.

Execute the following statements:

```
USE [master]
GO
CREATE USER [common_user] FOR LOGIN [common_user]
GO
USE [master]
GO
EXEC sp_addrolemember N'SqlJDBCXAUser', N'common_user'
GO
```

Setting Up Oracle

Configure the Oracle database as follows for use with SOA Systinet:

- If you intend to use the SOA Systinet Full Text Search feature, include the "Oracle Text" extension when installing the Oracle server. The "Oracle Text" extension is applied to Oracle by default.
- Create a database that uses the Unicode for Database Character Set (AL32UTF8) and, if required, an appropriate National Character Set. UTF-8 is the preferred encoding.
- If you set the default JVM language to a language other than English, you *must* use JDBC driver version 10.1.0.2. Otherwise it is not possible to deploy any SOA Systinet components.
- HP Software recommend setting the *cursor_sharing* parameter to `FORCE` to improve performance and economize shared pool usage.
- Create accounts based on the database access required by SOA Systinet during the installation.

  For details, see Database Installation Types on page 36.

  - For the Create Database an account is created by the installer.

- For the Create Schema option, if you want to separate the SOA Systinet data (recommended), create a tablespace in the database. Create a power user to own the schema, with the new tablespace as its default tablespace.

  For details, see Setting Up an Oracle Power User on page 44 .

- For Manual Database Arrangement, create a tablespace in the database, create a power user account to own the schema, with the new tablespace as its default tablespace. Create a *common user* account with minimal privileges.

  For details, see Setting Up an Oracle Power User on page 44 and Setting Up an Oracle Common User on page 44.

After installation, if you require SOA Systinet Full Text Search to be enabled, set up Oracle indexing.

For details, see Enabling Full Text Search on Oracle on page 150.

Setting Up an Oracle Power User

In order to use the Create Schema option during installation, the database administrator should create a *power user* with appropriate privileges to the database.

To set up a power user on Oracle:

1. Create an account that can create schema items.
2. Grant privileges to the account to connect to the database and create tables, indexes, and sequences.
3. (Optional) Grant the account the privilege to execute "CTXSYS"."CTX_DDL."

   This is a precondition for using SOA Systinet Full Text feature on the database.

Setting Up an Oracle Common User

In cases where the database administrator restricts access to the database to just select, insert, update, and delete operations, SOA Systinet requires a user with these privileges.

The SOA Systinet schema must exist prior to creating the common user.

To set up a common user on Oracle:

1 Save the following SQL statements to the `script.sql` file. These statements set the environment, grant rights and create synonyms. Run with parameters `power_user` and `common_user` with real values.

```
set pagesize 0;
set pagesize 0;
set line 200;
set verify off
set feedback off
spool ./grant.sql
SELECT 'GRANT INSERT, UPDATE, DELETE, SELECT ON ' || table_name || ' TO &2;' FROM user_tables;
SELECT 'GRANT SELECT ON ' || sequence_name || ' TO &2;' FROM user_sequences;
spool off
spool ./synonyms.sql
SELECT 'CREATE SYNONYM ' || table_name || ' FOR &1' || '.' || table_name || ';' FROM user_tables;
SELECT 'CREATE SYNONYM ' || sequence_name || ' FOR &1' || '.' || sequence_name || ';' FROM
user_sequences;
spool off
```

2 Connect to the database as power_user and process the `script.sql` file. This produces the scripts `grant.sql` and `synonyms.sql`. Then, run `grant.sql`.

```
sqlplus power_user/password@SID
-- generate grant and create synonym statements
@script.sql power_user common_user
-- execute grant.sql
@grant.sql
exit
```

3 As common_user, run `synonyms.sql`.

```
sqlplus common_user/password@SID
-- execute synonym.sql
@synonyms.sql
exit
```

Oracle with WebSphere

Configure the Oracle database to support WebSphere with XA transactions over Oracle datasources.

As user SYS, run the following commands on your Oracle server:

```
grant select on pending_trans$ to public;
grant select on dba_2pc_pending to public;
grant select on dba_pending_transactions to public;
grant execute on dbms_system to <user>;
```

Setting Up Application Servers

SOA Systinet is deployed to J2EE application servers. Each different application server must be setup prior to SOA Systinet installation.

The setup of each application server is explained in the following sections:

- Setting Up JBoss on page 46
- Setting Up Oracle Application Server on page 64
- Setting Up WebLogic on page 75
- Setting Up WebSphere on page 89

Setting Up JBoss

Deployment to JBoss requires less setup than installation to other J2EE servers. The SOA Systinet installation wizard automates deployment. Datasources and JMS are set up on the host JBoss servers and the SOA Systinet EAR file is deployed. The installer also creates a script for setting up the server environment and launching JBoss in simple deployment scenarios.

You may need to modify the JBoss application server for it to host SOA Systinet in production environments.

These modifications are covered in the following sections:

- Configuring JMS for JBoss on page 47

  Configure JBoss to use the Oracle or DB2 database instead of the default HSQLDB.

- Modifying the JBoss Run Script on page 55

  Configure the JBoss `run` script.

  If you are using the SOA Systinet `serverstart` script, you still have to edit the memory allocation in the `run` script.

- Setting the Datasource MaxPoolSize on page 56

  Increase the datasource maximum pool size for production deployments.

- Configuring JBoss Port Numbers on page 57

  If you set the SOA Systinet endpoint to use ports other than the default 8080 and 8443, you must enable these ports on JBoss.

- Deploying SOA Systinet to a JBoss Cluster on page 57

  Configure and deploy SOA Systinet to a JBoss cluster.

Configuring JMS for JBoss

JBoss uses JMS preconfigured for HSQLDB, which is sufficient for lightweight use in evaluation deployments. However, it has difficulty with large numbers of requests. For production deployments the JMS service should be configured to use a supported database.

SOA Systinet uses XA transactions. The application server transaction manager should be configured to have a minimum of 5 minutes for XA transaction timeout.

For details, refer to your application server documentation.

To change the JMS configuration, follow the relevant procedure for your deployment:

- Using DB2 DS in Non-Clustered Deployments on page 48
- Using DB2 DS in Clustered Deployments on page 49
- Using MSSQL DS in Non-Clustered Deployments on page 51

Using DB2 DS in Non-Clustered Deployments

To set up JBoss JMS to use the DB2 DS in non-clustered deployments:

1. Copy the DB2 JDBC drivers `db2jcc.jar` and `db2jcc_license_cu.jar`to `JBOSS_HOME/server/default/lib`.
2. Delete the file `JBOSS_HOME/server/default/deploy/hsqldb-ds.xml .`
3. Copy `JBOSS_HOME/docs/examples/jca/db2-ds.xml` to `JBOSS_HOME/server/default/deploy`.
4. In the new copy of `db2-ds.xml`, edit the `connection-url`, `user-name`, and `password` elements to match your local environment.
5. Change the value of the `driver-class` element to `com.ibm.db2.jcc.DB2Driver`.
6. Change the value of the `jndi-name` element from `DB2DS` to `DefaultDS`.
7. Add a `max-pool-size` element at the same level as `password`, `user-name`, and `driver-class`. Set the value of `max-pool-size` to the maximum number of concurrent working users plus the number of concurrent task executions.

   If you do not have an estimate of these numbers, set the `max-pool-size` to 100.

Example 1: Excerpt from db2-ds.xml

```
<datasources>
  <local-tx-datasource>
    <jndi-name>DefaultDS</jndi-name>
    <connection-url>jdbc:db2://dbserver:50000/database</connection-url>
    <driver-class>com.ibm.db2.jcc.DB2Driver</driver-class>
    <user-name>soa_account</user-name>
    <password>soa_password</password>
    <min-pool-size>5</min-pool-size>
    <max-pool-size>15</max-pool-size>
    <metadata>
      <type-mapping>DB2</type-mapping>
    </metadata>
  </local-tx-datasource>
</datasources>
```

8 Save `db2-ds.xml`.

9 Delete the file `JBOSS_HOME/server/default/deploy/jms/hsqldb-jdbc2-service.xml`.

10 Copy `JBOSS_HOME/docs/examples/jms/db2-jdbc2-service.xml` into the folder `JBOSS_HOMEserver/default/deploy/jms`.

11 In the new copy of `db2-jdbc2-service.xml`, replace the string `DB2DS` with `DefaultDS`.

12 Save `db2-jdbc2-service.xml`.

13 Open `JBOSS_HOME/server/default/deploy/jms/jms-ds.xml` and set the `max-pool-size` element to the maximum number of parallel served execution requests.

14 Open `JBOSS_HOME/server/default/deploy/jboss-web.deployer/server.xml` and set the `maxThreads` attribute to the maximum number of parallel served users.

Using DB2 DS in Clustered Deployments

To set up JBoss JMS to use the DB2 DS in clustered deployments:

1 Copy the DB2 JDBC drivers `db2jcc.jar` and `db2jcc_license_cu.jar`to `JBOSS_HOME/server/`*`node1`*`/lib`.

node1 in the path refers to a copy of the `all` configuration folder.

2 Delete the file `JBOSS_HOME/server/default/node1/hsqldb-ds.xml` .

3 Copy `JBOSS_HOME/docs/examples/jca/db2-ds.xml` to `JBOSS_HOME/server/node1/deploy`.

4 In the new copy of `db2-ds.xml`, edit the `connection-url`, `user-name`, and `password` elements to match your local environment.

5 Change the value of the `driver-class` element to `com.ibm.db2.jcc.DB2Driver`.

6 Change the value of the `jndi-name` element from `DB2DS` to `DefaultDS`.

7 Add a `max-pool-size` element at the same level as `password`, `user-name`, and `driver-class`. Set the value of `max-pool-size` to the maximum number of concurrent working users plus the number of concurrent task executions.

If you do not have an estimate of these numbers, set the `max-pool-size` to 100.

For an example of `db2-ds.xml`, see Step 7 in Using DB2 DS in Non-Clustered Deployments on page 48.

8 Save `db2-ds.xml`.

9 Delete the file `JBOSS_HOME/server/node1/deploy-hasingleton/jms/hsqldb-jdbc2-service.xml`.

10 Copy `JBOSS_HOME/docs/examples/jms/db2-jdbc2-service.xml` into the folder `JBOSS_HOME/server/node1/deploy-hasingleton/jms`.

11 In the new copy of `db2-jdbc2-service.xml`, replace the string `DB2DS` with `DefaultDS`.

12 Save `db2-jdbc2-service.xml`.

13 Open `JBOSS_HOME/server/node1/deploy/jms/hajndi-jms-ds.xml` and set the `max-pool-size` element to the maximum number of parallel served execution requests.

14 Open JBOSS_HOME/server/*node1*/deploy/jboss-web.deployer/server.xml and set the maxThreads attribute to the maximum number of parallel served users.

Using MSSQL DS in Non-Clustered Deployments

To set up JBoss JMS to use the MSSQL DS in non-clustered deployments:

1 Copy the MSSQL JDBC driver sqljdbc.jar to JBOSS_HOME/server/default/lib.

2 Delete the file JBOSS_HOME/server/default/deploy/hsqldb-ds.xml .

3 Copy JBOSS_HOME/docs/examples/jca/mssql-ds.xml to JBOSS_HOME/server/default/deploy.

4 In the new copy of mssql-ds.xml, edit the connection-url, user-name and password elements to match your local environment.

5 Change the value of the jndi-name element from MSSQLDS to DefaultDS.

6 Add a max-pool-size element at the same level as password, user-name, and driver-class. Set the value of max-pool-size to the maximum number of concurrent working users plus the number of concurrent task executions.

If you do not have an estimate of these numbers, set the max-pool-size to 100.

7 Save mssql-ds.xml.

8 Delete the file JBOSS_HOME/server/default/deploy/jms/hsqldb-jdbc2-service.xml.

9 Copy JBOSS_HOME/docs/examples/jms/mssql-jdbc2-service.xml into the folder JBOSS_HOME/server/default/deploy/jms

10 In the new copy of mssql-jdbc2-service.xml, replace the string MSSQLDS with DefaultDS.

11 Save mssql-jdbc2-service.xml.

12 Open JBOSS_HOME/server/default/deploy/jms/jms-ds.xml and set the max-pool-size element to the maximum number of parallel served execution requests.

13 Open JBOSS_HOME/server/default/deploy/jboss-web.deployer/server.xml and set the maxThreads attribute to the maximum number of parallel served users.

Using MSSQL DS in Clustered Deployments

To set up JBoss JMS to use the MSSQL DS in clustered deployments:

1. Copy the MSSQL JDBC driver `sqljdbc.jar` to `JBOSS_HOME/server/`*`node1`*`/lib`.

   > *`node1`* in the path refers to a copy of the `all` configuration folder.

2. Delete the file `JBOSS_HOME/server/`*`node1`*`/deploy/hsqldb-ds.xml` .
3. Copy `JBOSS_HOME/docs/examples/jca/mssql-ds.xml` to `JBOSS_HOME/server/`*`node1`*`/deploy`.
4. In the new copy of `mssql-ds.xml`, edit the `connection-url`, `user-name` and `password` elements to match your local environment.
5. Change the value of the `jndi-name` element from `MSSQLDS` to `DefaultDS`.
6. Add a `max-pool-size` element at the same level as `password`, `user-name`, and `driver-class`. Set the value of `max-pool-size` to the maximum number of concurrent working users plus the number of concurrent task executions.

   If you do not have an estimate of these numbers, set the `max-pool-size` to 100.
7. Save `mssql-ds.xml`.
8. Delete the file `JBOSS_HOME/server/`*`node1`*`/deploy-hasingleton/jms/hsqldb-jdbc2-service.xml`.
9. Copy `JBOSS_HOME/docs/examples/jms/mssql-jdbc2-service.xml` into the folder `JBOSS_HOME/server/`*`node1`*`/deploy-hasingleton/jms`
10. In the new copy of `mssql-jdbc2-service.xml`, replace the string `MSSQLDS` with `DefaultDS`.
11. Save `mssql-jdbc2-service.xml`.
12. Open `JBOSS_HOME/server/d`*`node1`*`/deploy/jms/hajndi-jms-ds.xml` and set the `max-pool-size` element to the maximum number of parallel served execution requests.

13 Open JBOSS_HOME/server/*node1*/deploy/jboss-web.deployer/server.xml and set the maxThreads attribute to the maximum number of parallel served users.

Using Oracle DS in Non-Clustered Deployments

To set up JBoss JMS to use the Oracle DS in non-clustered deployments:

1 Copy the Oracle JDBC driver ojdbc14.jar to JBOSS_HOME/server/default/lib.

2 Delete the file JBOSS_HOME/server/default/deploy/hsqldb-ds.xml .

3 Copy JBOSS_HOME/docs/examples/jca/oracle-ds.xml to JBOSS_HOME/server/default/deploy.

4 In the new copy of oracle-ds.xml, edit the connection-url, user-name, and password elements to match your local environment.

5 Change the value of the jndi-name element from OracleDS to DefaultDS.

6 Add a max-pool-size element at the same level as password, user-name, and driver-class. Set the value of max-pool-size to the maximum number of concurrent working users plus the number of concurrent task executions.

If you do not have an estimate of these numbers, set the max-pool-size to 100.

7 Save oracle-ds.xml.

8 Delete the file JBOSS_HOME/server/default/deploy/jms/hsqldb-jdbc2-service.xml.

9 Copy JBOSS_HOME/docs/examples/jms/oracle-jdbc2-service.xml into the folder JBOSS_HOME/server/default/deploy/jms

10 In the new copy of oracle-jdbc2-service.xml, replace the string OracleDS with DefaultDS.

11 Save oracle-jdbc2-service.xml.

12 Open JBOSS_HOME/server/default/deploy/jms/jms-ds.xml and set the max-pool-size element to the maximum number of parallel served execution requests.

13 Open JBOSS_HOME/server/default/deploy/jboss-web.deployer/server.xml and set the maxThreads attribute to the maximum number of parallel served users.

Using Oracle DS in Clustered Deployments

To set up JBoss JMS to use the Oracle DS in clustered deployments:

1. Copy the Oracle JDBC driver `ojdbc14.jar` to `JBOSS_HOME/server/`*`node1`*`/lib`.

   `node1` in the path refers to a copy of the `all` configuration folder.

2. Delete the file `JBOSS_HOME/server/`*`node1`*`/deploy/hsqldb-ds.xml` .
3. Copy `JBOSS_HOME/docs/examples/jca/oracle-ds.xml` to `JBOSS_HOME/server/`*`node1`*`/deploy`.
4. In the new copy of `oracle-ds.xml`, edit the `connection-url`, `user-name`, and `password` elements to match your local environment.
5. Change the value of the `jndi-name` element from `OracleDS` to `DefaultDS`.
6. Add a `max-pool-size` element at the same level as `password`, `user-name`, and `driver-class`. Set the value of `max-pool-size` to the maximum number of concurrent working users plus the number of concurrent task executions.

   If you do not have an estimate of these numbers, set the `max-pool-size` to 100.
7. Save `oracle-ds.xml`.
8. Delete the file `JBOSS_HOME/server/`*`node1`*`/deploy-hasingleton/jms/hsqldb-jdbc2-service.xml`.
9. Copy `JBOSS_HOME/docs/examples/jms/oracle-jdbc2-service.xml`into the folder `JBOSS_HOME/server/`*`node1`*`/deploy-hasingleton/jms`
10. In the new copy of `oracle-jdbc2-service.xml`, replace the string `OracleDS` with `DefaultDS`.
11. Save `oracle-jdbc2-service.xml`.
12. Open `JBOSS_HOME/server/d`*`node1`*`/deploy/jms/hajndi-jms-ds.xml` and set the `max-pool-size` element to the maximum number of parallel served execution requests.

13 Open JBOSS_HOME/server/*node1*/deploy/jboss-web.deployer/server.xml and set the maxThreads attribute to the maximum number of parallel served users.

Modifying the JBoss Run Script

When you launch SOA Systinet with the SOA_HOME/bin/serverstart script, it sets up the JBoss environment before calling the JBoss run script. No further setup is necessary for most evaluation or development scenarios. However, serverstart is not appropriate for all production environments.

The following sections describes how to alter the JBoss run script for use in production deployments:

- Setting awt.headless on page 55
- JBoss Memory Allocation on page 55

Setting awt.headless

If JBoss is installed on UNIX, set the java.awt.headless property to "true."

To set java.awt.headless:

1 Open the JBOSS_HOME/bin/run.bat|run.conf script in an editor.

2 Insert this line where JAVA_OPTS is set:

```
-Djava.awt.headless=true
```

3 Save and exit the script.

JBoss Memory Allocation

Increase the maximum memory limit on the JBoss server to optimize SOA Systinet performance.

To change the memory settings:

1 Open the run script in the bin directory of the JBoss server. On Windows this script is run.bat and on UNIX systems this is run.conf.

2 Find the following lines:

```
rem JVM memory allocation pool parameters. Modify as appropriate.
            set JAVA_OPTS=%JAVA_OPTS% -Xms128m...
```

3 Edit the lines as follows:

```
rem JVM memory allocation pool parameters. Modify as appropriate.
            set JAVA_OPTS=%JAVA_OPTS% -Xms128m -Xmx1300m -XX:MaxPermSize=256m
```

If you are using the `serverstart` script to launch JBoss, you can delete this line from the JBoss `run` script instead of editing it. If you leave it, however, it will overwrite the memory allocation parameters set by `serverstart`.

4 Save and exit the script.

Memory sizing should take performance requirements into consideration for the deployed system. These settings are only a recommendation.

Setting the Datasource MaxPoolSize

The default JBoss datasource Maximum Pool Size is not adequate for a production environment. The default MaxPoolSize based on default Oracle configuration is only 15, for example. The Maximum Pool Size should be at least 1/4 the number of parallel requests that you require to be handled simultaneously.

To increase the maximum pool size:

1 Open `JBOSS_HOME/server/CONFIG_HOME/deploy/hpsoasystinet-xa-ds.xml` in an editor. (CONFIG_HOME refers to the JBoss configuration to which you have deployed SOA Systinet. For non-clustered deployments, this is usually `default` and for clustered deployments, this is usually `all`.)

2 Edit the element `max-pool-size`. Its value should be at least 1/4 of the number of simultaneous parallel requests.

3 Save your changes and exit.

Configuring JBoss Port Numbers

By default, SOA Systinet uses ports 8080 and 8443. If you select a different set of ports during installation, you must configure JBoss after installation to use these ports. If you are using port numbers that are higher than the default, the easiest way is to edit the JBoss configuration files:

To edit the JBoss port numbers:

1. Open `JBOSS_HOME/server/CONFIG_HOME/conf/jboss-service.xml` in an editor. (CONFIG_HOME refers to the JBoss configuration to which you have deployed SOA Systinet. For non-clustered deployments, this is usually `default` and for clustered deployments, this is usually `all`.)

2. Search for the string `ports-01`. The search function takes you to the following commented-out MBean:

```
<!-- (comment text)......
<mbean code="org.jboss.services.binding.ServiceBindingManager"
          name="jboss.system:service=ServiceBindingManager">
    <attribute name="ServerName">ports-01</attribute>
    <attribute name="StoreURL">
          ${jboss.home.url}/docs/examples/binding-manager/sample-bindings.xml
    </attribute>
    <attribute name="StoreFactoryClassName">
          org.jboss.services.binding.XMLServicesStoreFactory
    </attribute>
</mbean>
          -->
```

3. Remove the wrapping comment tag and comment text from the MBean.

4. Set the value of the element `<attribute name="ServerName">ports-01</attribute>`. This value represents the factors of 100 by which additional port numbers above the default value are enabled. For example, if you leave the value at `ports-01`, ports 8180, 8280, 8380... are enabled. If you set the value at `ports-02`, the additional ports are 8280, 8480, 8680...

5. Save your changes and exit the editor.

Deploying SOA Systinet to a JBoss Cluster

SOA Systinet installers do not support clustered installation. It is therefore necessary to install to a temporary standalone JBoss configuration and then manually create cluster nodes from that standalone JBoss.

This section gives instructions for creating a cluster JBoss application servers where each hosts one JBoss instance with SOA Systinet. A JBoss instance with a configuration named `node1` is located on the first machine, `node2` on the second one, and additional nodes as required. A load balancer runs on the first machine.

For details, see the following sections:

- Load Balancing on JBoss on page 58
- Installing SOA Systinet to a JBoss Cluster on page 61

Load Balancing on JBoss

The following instructions are for the use of the `mod_jk` module in Apache 2.2 but you can use any passive-cookie load balancer which is supported by JBoss. For more information about `mod_jk`, see the Apache documentation [http://tomcat.apache.org/tomcat-3.3-doc/mod_jk-howto.html]. You can download `mod_jk` from the Apache site [http://tomcat.apache.org/connectors-doc/]. There is also a version you can copy and paste in Example 2 on page 59.

Example 2: Pasteable mod_jk.conf

```
# Load mod_jk module
# Specify the filename of the mod_jk lib
LoadModule jk_module modules/mod_jk-apache-2.2.3.so

# Where to find workers.properties
JkWorkersFile conf/workers.properties

# Where to put jk logs
JkLogFile logs/mod_jk.log

# Set the jk log level [debug/error/info]
JkLogLevel info

# Select the log format
JkLogStampFormat  "[%a %b %d %H:%M:%S %Y]"

# JkOptions indicates to send SSK KEY SIZE
JkOptions +ForwardKeySize +ForwardURICompat -ForwardDirectories

# JkRequestLogFormat
JkRequestLogFormat "%w %V %T"

# Mount your applications
JkMount /* loadbalancer

# You can use external file for mount points.
# It will be checked for updates each 60 seconds.
# The format of the file is: /url=worker
# /examples/*=loadbalancer
JkMountFile conf/uriworkermap.properties

# Add shared memory.
# This directive is present with 1.2.10 and
# later versions of mod_jk, and is needed for
# for load balancing to work properly
JkShmFile logs/jk.shm

# Add jkstatus for managing runtime data
<Location /jkstatus/>
    JkMount status
    Order deny,allow
    Deny from all
    Allow from 127.0.0.1
```

```
</Location>
```

To set up `mod_jk` load balancing:

1. Install an Apache server, or configure an existing Apache server, to use the ports and host name which will be used for SOA Systinet. Also configure SSL if it is required for deployment.
2. Copy `mod_jk.conf` to `APACHE/conf`.
3. In the Apache Tomcat `/conf` directory, edit `httpd.conf`. Add the line `Include conf/mod_jk.conf` to the end of the file. Make other changes to `httpd.conf` as described in that file's comments and in the Apache documentation.
4. Modify contexts in the file `APACHE\conf\uriworkermap.properties`, if necessary.
5. Modify workers settings in the file `APACHE\conf\workers.properties`. Change `worker.`*`nodeName`*`.port`, `worker.`*`nodeName`*`.host`, `worker.loadbalancer.balance_workers` and the number of workers. Names of nodes (*`nodeName`*) must match names of corresponding JBoss configurations. Example 3 on page 61 is a modified `workers.properties` file.
6. Run the Apache server with the configured load balancer.

Example 3: Modified workers.properties

```
# Define list of workers that will be used
# for mapping requests
worker.list=loadbalancer,status

# Define Node1
# modify the host as your host IP or DNS name.
worker.node1.port=8009
worker.node1.host=server1
worker.node1.type=ajp13
worker.node1.lbfactor=1

# Define Node2
# modify the host as your host IP or DNS name.
worker.node2.port=8009
worker.node2.host=server2
worker.node2.type=ajp13
worker.node2.lbfactor=1

# Load-balancing behaviour
worker.loadbalancer.type=lb
worker.loadbalancer.balance_workers=node1,node2
worker.loadbalancer.sticky_session=1

# Status worker for managing load balancer
worker.status.type=status
```

Installing SOA Systinet to a JBoss Cluster

Install SOA Systinet to a temporary JBoss configuration.

To install SOA Systinet when clustering JBoss:

1. Copy the `JBOSS_HOME/server/all` configuration. Name the copy `nodeX`.
2. Launch the SOA Systinet installer.
3. In the **Endpoint Properties** step, pass the load balancer hostname and endpoints.

4 In the **Application Server Properties** step, deploy to the `nodeX` configuration.

When the load balancer is running and SOA Systinet is installed to a temporary configuration, create the first node to use for running SOA Systinet.

To create the first node:

1 Copy the `JBOSS_HOME/server/all` configuration. Name the copy `node1`.

2 Set up JMS for your database.

For details, see Configuring JMS for JBoss on page 47.

3 Copy the `nodeX` datasource, `JBOSS_HOME\server\nodeX\deploy\hpsoasystinet-xa-ds.xml`. Paste it into `JBOSS_HOME\server\node1\deploy\`

4 Copy the `nodeX` mail configuration, `JBOSS_HOME\server\nodeX\deploy\mail-service.xml`. Paste it into `JBOSS_HOME\server\node1\deploy\`

5 Enable the use of the `mod_jk` load balancer. Set the value of the `UseJK` attribute to `true` in the file `JBOSS_HOME\server\node1\deploy\jboss-web.deployer\META-INF\jboss-service.xml`

6 Open the file `JBOSS_HOME\server\node1\deploy\jboss-web.deployer\server.xml` for editing.

7 Comment out the HTTP connector listening at port 8080.

This step is optional, but an existing HTTP listener can hide a misconfiguration or a bug.

8 Add the attribute `jvmRoute="${jboss.server.name}"` to the `Engine` element with the name `jboss.web`. (Do not evaluate the attribute value. Place it in the configuration file as is. It will be evaluated by JBoss at runtime.) The `jvmRoute="${jboss.server.name}"` attribute appends a suffix with the node name to outgoing JSESSIONID headers. These suffixes are used by the load balancer to maintain session affinity.

9 Apply the following workaround to disable session replication among clusters:

a Open the file `JBOSS_HOME\server\node1\deploy\jboss-web-cluster.sar\META-INF\jboss-service.xml` for editing.

b Change the value of `buddyReplicationEnabled` from false to true.

Change the value of `numBuddies` from 1 to 0.

10 Copy the following files from `JBOSS_HOME\server\nodeX\conf` to `JBOSS_HOME\server\node1\conf`:

- `roles.properties`
- `users.properties`
- `server.cer`
- `server.keystore`

11 Create an additional cluster node.

a Copy your JBoss installation to a second computer.

b Create a `JBOSS_HOME/server/node2` directory on that computer.

c Copy the directory `JBOSS_HOME\server\node1` to the node2 directory on the second computer.

d Repeat for node3, node4...nodeN.

12 Copy the following files from `JBOSS_HOME\server\nodeX\deploy` to `JBOSS_HOME\server\node1\farm\`. They are distributed to all other cluster nodes when those nodes boot.

- hp-soa-systinet.ear
- hp-soa-systinet.sar

13 Launch `node1` on the first computer. When it successfully starts, launch `node2` node on the second computer. Continue for all other nodes. For each node, it is necessary to specify the URL of the HA-JNDI service in the local JBoss. Base the command-line for starting a node on the following:

```
JBOSS_HOME\bin\run.bat -b 0.0.0.0 -c nodeName
     -Dhpsoa.hajndi.url-jnp://hostname:1100/
     -Djboss.partition.name=DefaultPartition
```

14 SOA Systinet should be running on `http://balancerHostname:port/soa/`

Setting Up Oracle Application Server

SOA Systinet requires some initial configuration of OAS before you can deploy to it.

In this documentation, `OAS_HOME` refers to the installation directory for the Oracle Application Server.

The procedures for setting up OAS for SOA Systinet are described in the following sections:

- Update JDBC Driver on page 64
- Allocating Memory and Setting Java Options on page 65
- Creating Resources on page 67
- Setting Up SSL on OAS on page 72

Update JDBC Driver

The version of Oracle Application Server that SOA Systinet supports has an out-of-date JDBC driver. Therefore, you must download and install a new copy of `ojdbc14dms.jar`.

To update the JDBC driver for OAS:

1. Make sure OC4J is not running.
2. Open a command console in `OAS_HOME/product/10.1.3.1/oas1/jdbc/lib`.
3. Rename the existing `ojdbc14dms.jar` file to `ojdbc14dms.jar.backup`.
4. In your web browser, open the Oracle JDBC driver download page [http://www.oracle.com/technology/software/tech/java/sqlj_jdbc/htdocs/jdbc_10201.html] and download version 10.0.2.0 or later of `ojdbc14dms.jar`.
5. Copy the newly downloaded `ojdbc14dms.jar` to `OAS_HOME/product/10.1.3.1/oas1/jdbc/lib`.

Allocating Memory and Setting Java Options

SOA Systinet requires the following changes to the Oracle Application Server memory and Java settings:

- Increase the memory maximum permanent size to 512MB.
- Increase the memory initial heap size to 800MB.
- Increase the memory maximum heap size to at least 1000MB, depending on your deployment.
- Set the Java property, `-Djava.awt.headless=true`.
- Set the Java property, `-Doc4j.userThreads=true`.

You can make these changes with the Oracle administration console or by editing `opmn.xml`.

To edit the settings with the console:

1. Start the Administration Console for the OC4J instance that will host SOA Systinet.
2. Open the Administration tab.
3. From the menu, select **Administration Tasks→Properties→Server Properties→Command Line Options**.
4. Set the Initial Heap Size to `800M`.
5. Set the Maximum Heap Size to at least `1000M`.
6. Add or reset the following Java options:
   - `-XX:MaxPermSize=512M`
   - `-Djava.awt.headless=true`
   - `-Doc4j.userThreads=true`
7. Click **Apply**.

8 Restart the OC4J instance.

To edit the settings in `opmn.xml`:

1 Make sure the OC4J instance is not running.

2 Open `OAS_HOME/opmn/conf/opmn.xml` with a text editor.

3 Modify or add the following settings to the start parameters:

- `-XX:MaxPermSize=512M`
- `-Xms800M`
- `-Xmx1000M`
- `-Djava.awt.headless=true`
- `-Doc4j.userThreads=true`

Memory sizing should take performance requirements into consideration for the deployed system. These settings are only a recommendation.

4 Save your changes.

5 Restart the OC4J instance.

Example 4: opmn.xml Example

```
<?xml version = '1.0' encoding = 'UTF-8'?>
<opmn xmlns="http://www.oracle.com/ias-instance">
  ...
  <process-manager>
    ...
    <ias-instance id="server1.domain.com" name="server1.domain.com">
      ...
      <ias-component id="default_group">
        <process-type id="home" module-id="OC4J" status="enabled">
          <module-data>
            <category id="start-parameters">
              <data id="java-options" value="-Xrs -server -XX:MaxPermSize=512M -Xms800M -Xmx1000M
                  -XX:AppendRatio=3 -Djava.security.policy=$ORACLE_HOME/j2ee/home/config/java2.policy
                  -Djava.awt.headless=true -Dhttp.webdir.enable=false -Doc4j.userThreads=true"/>
            </category>
            ...
          </module-data>
          ...
        </process-type>
      </ias-component>
    </ias-instance>
  </process-manager>
</opmn>
```

Creating Resources

SOA Systinet requires JDBC and JMS resources to communicate with the database.

In OAS, use the Application Server Control to create them.

To use the Application Server Control to create resources:

1. Start OAS with the following command:

   OAS_HOME/opmn/bin/opmnctl startall

2. Open the Application Server Control (`http://localhost/em`).

3. Log in as the OC4J administrator.

4 Click the link to your OC4J instance (by default named Home) in the tree view on the main page.

5 Open the Administration tab.

Use the Administration tab to create JDBC and JMS resources.

For details, see the following sections:

- Creating JDBC Resources on page 68
- Creating JMS Resources on page 70

Creating JDBC Resources

SOA Systinet requires a global JDBC datasource to communicate with the database. Before creating this datasource, you must create a connection pool.

SOA Systinet uses XA transactions. The application server transaction manager should be configured to have a minimum of 5 minutes for XA transaction timeout.

For details, see your application server documentation.

In the Administration tab, create a connection pool and then a datasource that uses the connection pool.

To create a connection pool:

1 From the menu, select **Administration Tasks→Services→JDBC Resources**.

2 In the Connection Pools section, click **Create**.

The Create Connection Pool - Application wizard opens.

3 Leave Application as **default**, select **New connection pool**, and then click **Continue**.

A details page for a new connection pool opens.

4 Name the connection pool `HP SOA Systinet Connection Pool`.

5 Keep the default Connection Factory Class (`oracle.jdbc.pool.OracleDataSource`).

6 Select **Generate URL From Connection Configuration** and enter the following connection property fields:

Field name	Value
Driver Type	Thin
DB Host Name	*Type the database host name*
DB Listener Port	*Type the database port*
DB Identifier Type	SID
SID/Service Name	*Type sid*
Username (under Credentials section)	*Type your database username*
Use Cleartext Password / Password	*Type your database password*

Use the same database parameters for SOA Systinet installation.

7 Click **Test Connection**. This tests the connection properties you entered in Step 6. Check the connection properties if the test fails.

8 Open the Attributes tab and enter the following parameters:

Field name	Value
Initial size of Connection Cache	5
Minimum Number of Connections	5
Maximum Number of Connections	200
Validate Connection	True

9 Click **Finish**.

After you create a connection pool, you can use it to create a datasource.

To create a datasource:

1 From the manu, select **Administration Tasks→Services→JDBC Resources**.

2 In the Data Sources section, click **Create**.

The Create Data Source - Application & Type wizard opens.

3 Leave **Application** as **default**, select **Managed data source**, and then click **Continue**.

A details page for a new datasource opens.

4 Enter the following parameters:

Parameter	Value
Name	HP SOA Systinet Data Source
JNDI Location	hpsoasystinetDS
Transaction Level	Global and Local Transactions
Connection Pool	HP SOA Systinet Connection Pool
Login Timeout	0 *(default)*

Leave the credentials fields empty. The connection pool credentials will be used.

5 Click **Finish**.

Creating JMS Resources

SOA Systinet requires connection factories and destinations.

You can configure JMS to meet your requirements. This section describes a JMS setup that ensures that JMS resources are accessible by SOA Systinet and function correctly.

To create JMS connection factories for SOA Systinet:

1 From the menu, select **Administration Tasks→Services→Enterprise Messaging Service→JMS Connection Factories**.

2 Click **Create New**.

A wizard opens for creating a connection factory.

3 Create the following connection factories. Each row corresponds to one connection factory. Leave blank all fields not included in the following table. After you enter the values for a connection factory, click **OK**.

Repeat from Step 2 until you have created all the connection factories.

Connection Factory Type	JNDI Location	Check "XA Enabled"?
Queue	jms/ReportingSenderConnectionFactory	Yes
Queue	jms/ReportingReceiverConnectionFactory	Yes
Queue	jms/PMConnectionFactory	Yes
Queue	jms/platform/QueueConnectionFactory	Yes
Topic	jms/platform/TopicConnectionFactory	Yes

To create JMS destinations for SOA Systinet:

1 From the menu, select **Administration Tasks→Services→Enterprise Messaging Service→JMS Destinations**.

2 Click **Create New**.

A wizard opens for creating a JMS destination.

3 Create the following destinations. Each row corresponds to one destination. Leave blank all fields not included in the following table. After you enter the values for a destination, click **OK**.

Repeat from Step 2 until you have created all the destinations.

Destination Type	Destination Name	Select Persistence	JNDI Location	Persistence File
Queue	RF Executions Queue	File Based	queue/ReportingExecutions	RF_ReportingExecutionsQueue
Queue	Platform scheduleTimerQueue	File Based	queue/scheduleTimerQueue	platform_scheduleTimerQueue
Queue	Platform taskProcessorQueue	File Based	queue/taskProcessorQueue	platform_taskProcessorQueue
Topic	Platform taskStopperTopic	File Based	topic/taskStopperTopic	platform_taskStopperTopic
Queue	PM Validation Queue	File Based	queue/Validation	PM_validationQueue

Setting Up SSL on OAS

The Oracle HTTP server contains a demo certificate. To use SSL communication for SOA Systinet, you must create a real certificate for OAS to trust.

This is an optional step, you can use the demo certificate if it meets the requirements for applications that connect to SOA Systinet.

To create a certificate for use by the Oracle HTTP Server:

1 Launch the Oracle Wallet Manager in one of the following ways:

- On Windows, select *`your_server_name`* **Integrated Management Tools→Wallet Manager** from the GUI menu.
- On UNIX systems, launch Wallet Manager with the `own` script in `OAS_HOME/bin`.

2 In the Wallet Manager GUI menu, select **Wallet→New**.

The Create New Wallet wizard opens.

3 If you are asked to create a default wallet directory, select **No**.

4 Select **Standard** wallet type.

5 Enter a password for the wallet.

This password must be alphanumeric, with at least 8 characters.

6 Click **OK**.

An empty wallet is created.

7 When prompted to create a certificate for the wallet, click **Yes**.

A detail page for the certificate opens.

8 For the certificate common name (CN), enter the hostname of the OAS server.

The hostname must be written exactly as it appears in the URL of the OAS server.

9 Enter other details, according to your company policy on creating certificates.

10 From the menu, select **Wallet→Save**.

When prompted, enter the following location:

`OAS_HOME\Apache\Apache\conf\ssl.wlt\new`.

Confirm that you want to create this directory.

11 From the menu, select **Wallet→Auto Login**.

12 From the menu, select **Wallet→Save**.

13 Export the certificate request in one of the following ways:

a. From the menu, select **Operations→Export Certificate Request**.

b. You are prompted for the type of certificate file and the location.

   Select the CSR file type and the location of your choice.

14. Exit Wallet Manager.

15. Request a Certification Authority (CA) to sign your exported certificate.

    It may be signed by one of the following authorities:

    - A public certification authority, such as VeriSign.
    - A corporate certification authority. Check your company IT security guidelines.
    - Yourself, with the OpenSSL tool.

16. When you obtain a signed copy of your certificate from the CA, you should also obtain a copy of the CA's own certificate.

17. Reopen Wallet Manager and the wallet you created.

18. From the menu, select **Operations→Import Trusted Certificate**.

    Import the certificate of the CA that signed your certificate.

19. From the menu, select **Operations→Import User Certificate**.

    Import the signed copy of your certificate.

20. From the menu, select **Wallet→Save**, and then exit Wallet Manager.

21. Set up Oracle HTTP server to use the wallet you created.

    - Open `OAS_HOME\Apache\Apache\conf\ssl.conf` in an editor.
    - Change the `SSLWallet` directive to point to the newly generated wallet file at `OAS_HOME\Apache\Apache\conf\ssl.wlt\new`.

Setting Up WebLogic

WebLogic requires initial configuration before you can deploy SOA Systinet to it.

In this documentation, WL_HOME refers to the WebLogic server installation directory.

WebLogic setup is described in the following sections:

Creating Domains for SOA Systinet

You must host SOA Systinet in a separate WebLogic domain.

To create a WebLogic domain:

1. Launch the WebLogic Configuration wizard with the following command:

   WL_HOME/common/bin/config

2. Select **Create a new WebLogic Domain**, and click **Next**.
3. The default settings for the domain source should not require any changes. Click **Next**.
4. Enter the WebLogic administrator username and password, and click **Next**.
5. In WebLogic Domain Startup Mode, select **Development** or **Production**.
6. In JDK Selection, select **Sun SDK**, and click **Next**.
7. In the Customize Environments and Settings page, do one of the following:

- Select **No** for a single server deployment with default settings and continue this procedure.
- Select **Yes** for clustered deployment.

8 Enter a name and location for the domain.

In this documentation, the domain location is referred to as `DOMAIN_HOME`, and the name as *`hpsoa_domain`*.

9 Click **Create**.

WebLogic creates the new domain in the specified location.

10 Click **Done** to exit the wizard.

Setting Up the WebLogic Server

You must create a WebLogic server to host SOA Systinet.

To set up the WebLogic server:

1 Start the WebLogic Node Manager with the following command:

WL_HOME/server/bin/startNodeManager

WebLogic 10.0 node manager uses system variables *`PATH`* and *`CLASSPATH`* in the server start command. The node manager does not handle these variables if they contain spaces.

To avoid this problem, do one of the following:

- On Windows, replace the conflicting parts of the paths with DOS-like 8.3 file names and restart node manager.
- Edit `WL_HOME/common/nodemanager/nodemanager.properties`, and add the parameter *`StartScriptEnabled=true`*, and then restart node manager.

2 Start the WebLogic server for your domain with the following command:

DOMAIN_HOME/startWebLogic

3 In your browser, open the WebLogic Administration Console:

`http://localhost:7001/console`

4 Log in with the administrator credentials created in Creating Domains for SOA Systinet on page 75.

5 In the web console, click **Lock & Edit**.

6 In the Domain Structure section, expand **Services** and select **JTA**.

The domain Settings page opens.

7 HP Software recommend setting **Timeout Seconds** to 300 and clicking **Save**.

8 In the Domain Structure section, expand **Environment** and select **Machines**.

The Summary of Machines page opens.

9 Click **New**.

The Create a New Machine page opens.

10 Input a name, for example HP SOA Machine, and then click **OK**.

If you use a UNIX operating system, set the Machine OS to **Unix**.

11 In the Domain Structure section, expand **Environment** and select **Servers**.

The Summary of Servers page opens.

12 Click **New**.

The Create a New Server page opens.

13 Input the Server Name and the Server Listen Port, for example `HP_SOA_Server` and `7003`, and then click **Finish**.

The port must not be in use.

14 In the Summary of Servers page, click the new server name.

The Settings page opens.

15 Set **Machine** to the value used in Step 10.

16 Select **SSL Listen Port Enabled** and input the SSL Listener Port, for example 7004, and then click **Save**.

The port must not be in use.

17 Select the Control tab, and click **Start**, and then click **Yes**.

18 In the Change Center section, click **Activate Changes**.

Creating Resources

SOA Systinet requires a number of resources to be setup in WebLogic.

Use the Administration Console to create them.

To use the Administration Console to create resources:

1 In your browser, open the WebLogic Administration Console:

```
http://localhost:7001/console
```

2 Log in as the WebLogic administrator.

3 Click **Lock & Edit**.

4 Use the Administration Console to create mail, JDBC, and JMS resources.

For details, see the following sections:

- Creating Mail Sessions on page 79
- Creating JDBC Resources on page 80
- Creating JMS Resources on page 84

5 In the Change Center section, click **Activate Changes** to apply your changes.

You can verify your resources setup in your browser:

```
http://localhost:7001/console/consolejndi.portal
```

Creating Mail Sessions

SOA Systinet requires a mail session for automated notifications.

To create an SOA Systinet Mail Session:

1 In the Domain Structure section, expand **Services** and select **Mail Sessions**.

The Summary of Mail Sessions page opens.

2 Click **New**.

The Create a New Mail Session page opens.

3 Enter a name, and then click **OK**.

4 In the Summary of Mail Sessions page, click the new mail session name.

5 Enter the JNDI Name, `/Mail`, enter JavaMail Properties according to your environment mail settings, for example, `mail.transport.protocol=smtp;mail.user=builder;mail.smtp.host=mail.com;mail.debug=true`), and then click **Save**.

6 In the Settings page, select the **Targets** tab.

7 Target all servers and clusters hosting SOA Systinet, and click **Save**.

Creating JDBC Resources

SOA Systinet requires two JDBC datasources, an XA-enabled datasource and a non-XA-enabled datasource. These datasources handle all traffic between the SOA Systinet on WebLogic and the database server. Each WebLogic managed server and/or cluster server requires a persistent store on the database, which uses the non-XA-enabled datasource for communication.

Use the WebLogic Administration Console to create JDBC datasources. The Administration Server must be running.

SOA Systinet uses XA transactions. The application server transaction manager should be configured to have a minimum of 5 minutes for XA transaction timeout.

For details, see your application server documentation.

To create an XA-enabled JDBC datasource:

1 Open the Summary of JDBC Datasources page (Services→JDBC→Data Sources). Click **Lock and Edit** then **New**.

2 Give the datasource a unique, arbitrary descriptive name, such as HP SOA Systinet DS

3 Give the datasource the JNDI name hpsoasystinetDS.

JNDI names must be exact.

4 From the Database drop-down list, select the same database type that you use for SOA Systinet.

5 From the Database Driver drop-down list, select an XA-supporting JDBC database driver for the database type.

If you are using Oracle, select the Oracle "thin" XA driver.

6 Click **Next** to open the Transaction Options page and click Next again to open the Connection Properties page.

7 In the Connection Properties page, use the same database parameters you use for SOA Systinet.

Target all servers or clusters hosting SOA Systinet.

8 Click **Finish** to return to the Summary of JDBC Datasources page. and click **Activate Changes**.

The datasource you created appears in the table of datasources.

9 Click **Lock and Edit**.

10 Click the name of the XA datasource in the table of datasources to open its details page in the Configuration:General tab.

11 Open the Configuration:Connection Pool tab.

Increase the maximum capacity of the connection pool. The Maximum Capacity should be at least 1/4 the number of parallel requests that you require to be handled simultaneously. If you do not have an estimate of this number, set the maximum capacity to 100.

Click **Save** but do not activate changes. Remain in the Configuration:Connection Pool tab.

12 Increase the Initial Capacity to the number of expected concurrent users.

Click **Save** but do not activate changes.

13 Open the Configuration:Transactions tab and select **Use XA Datasource Interface**.

Click **Save** but do not activate changes.

14 Navigate out of the datasource details page, for example to the Summary of JDBC Datasources page, and click **Activate Changes**.

If you do not navigate out of the datasource details page before you save changes to the datasource, you cause a JDBCSystemResourceMBean cannot be null exception. This exception is harmless, because the changes to the datasource are activated anyway, but avoidable.

If the exception "Could not get JDBC Connection; nested exception is java.sql.SQLException: Internal error: Cannot obtain XAConnection weblogic.common.resourcepool.ResourceDisabledException: Pool hpsoasystinetDS is disabled, cannot allocate resources to applications..." occurs in the in log file, you can:

- Increase the count of connections in the datasource.
- Increase the timeout for acquiring a connection.

  Increase the value of Connection Reserve Timeout (Data Sources→hpsoasystinetDS→Connection Pool→Advanced). (Default is 10 seconds).

  Setting 0 (infinite waiting for connection) is not recommended because of a risk of deadlocks.

To create a non-XA-enabled JDBC datasource:

1 Open the JDBC datasources page (Services→JDBC→Data Sources). Click **Lock and Edit** and then **New**.

2 Give the datasource a unique, arbitrary descriptive name such as SOA Systinet JMS DS.

3 Give the datasource the JNDI name jms-hpsoasystinetDS.

JNDI names must be exact.

4 From the Database drop-down list, select the same database type that you use for SOA Systinet.

5 From the Database Driver drop-down list, select a non-XA-supporting JDBC database driver for the database type.

If you are using Oracle, select the Oracle "thin" non-XA driver.

6 Click Next to open the Transaction Options page.

7 Select **Supports Global Transactions** and click **Next** to open the Connection Properties page.

8 In the Connection Properties page, use the same database parameters you use for SOA Systinet.

Target all servers or clusters hosting SOA Systinet.

9 Click **Finish** and **Activate Changes**.

Create a JDBC persistent store for every migratable cluster server and every standalone server hosting SOA Systinet. These persistent stores use the `jms-hpsoasystinetDS` non-XA datasource.

To create a JDBC persistent store:

1 Navigate to Services→Persistent Stores.

2 Click **Lock and Edit**.

3 From the New drop-down menu, select **Create a JDBC Store** to open the Create JDBC Store wizard.

4 Give the persistent store a unique, arbitrary name, such as *`SERVER_NAME`* Store.

5 From the Target drop-down list, select the standalone managed server or migratable cluster server corresponding to the selected persistent store.

6 In the Datasource drop-down field, select the non-XA-enabled datasource you created.

7 Give the persistent store a unique prefix, so that the stores do not use the same table.

8 Click **Finish** and save your changes.

9 Repeat the procedure for each migratable cluster server and/or standalone managed server.

10 Click **Activate Changes**.

Creating JMS Resources

SOA Systinet requires JMS resources to be set up in WebLogic.

You can configure JMS to meet your requirements. This section describes a JMS setup that ensures that JMS resources are accessible by SOA Systinet and function correctly.

Create a JMS server for each migratable cluster server and each standalone server.

To create a JMS Server:

1 Open the JMS Servers page, Services→Messaging→JMS Servers.

2 Click **Lock and Edit** and then **New** to open the Create a New JMS Server wizard in the JMS Server Properties page.

3 In the Name field, give the JMS server a unique, arbitrary descriptive name, such as *SERVER_NAME* JMS, indicating which server is targeted.

4 From the Persistent Store drop-down field, select the persistent store of the managed server or migratable cluster server to target.

Click Next to open the Select Targets page

5 From the Target drop-down field, select the standalone managed server or migratable cluster server corresponding to the persistent store you selected.

Click **Finish** and save your changes.

Create a JMS Module that to contain definitions of JMS connection factories as well as required JMS destinations.

To create a JMS module:

1 Open the JMS Modules page, Services→Messaging→JMS Modules.

2 Click **Lock and Edit** and **New** to open the Create JMS System Module wizard.

3 Give the JMS module a unique, arbitrary descriptive name, such as HP SOA Systinet JMS Module.

You may apply any descriptor file name and location, or leave those fields blank to use the default.

Click **Next**.

4 Target the standalone server or the cluster hosting SOA Systinet.

5 Select **Would you like to add resources to this JMS system module?**, and click **Finish**.

The details page for the JMS Module opens in the Configuration tab.

6 Create the connection factories listed in Table 8.

Select the resource type and click **Next**. Leave blank all fields not included in the table. After you enter the values for a resource, click **Finish**.

Repeat for each connection factory.

Table 7. JMS Connection Factories for WebLogic

Name	JNDI Name
HP SOA Systinet Connection Factory	/ConnectionFactory
Reporting Sender Connection Factory	jms/ReportingSenderConnectionFactory
Reporting Receiver Connection Factory	jms/ReportingReceiverConnectionFactory

a Click **New** in the Summary of Resources table to open the Create a New JMS System Module Resource wizard.

b Select **Connection Factory** and click **Next** to open the Create a New Connection Factory wizard.

c Give the connection factory a unique, arbitrary descriptive name (for example the one listed in Table 7.)

d Give the connection factory the JNDI name specified in the JNDI Name column.

JNDI names must be exact.

e Use the default targeting, which selects the parent module target.

f Click **Finish** and **Activate Changes**.

g Edit each connection factory and select **XA Connection Factory Enabled** in the Configuration:Transactions tab.

h If the SOA Systinet host is a cluster, open the Configuration: Load Balancing tab and disable server affinity.

i Save your changes.

7 If the SOA Systinet host is a managed server and not a cluster create a subdeployment.

a Navigate to the SOA Systinet JMS Module details page.

b Open the Subdeployments tab and create a subdeployment for the JMS module.

c Set the subdeployment target as the SOA Systinet host managed server's JMS server.

d Create the resources listed in the Table 8.

Click **New** in the Resources table. Select the resource type and click **Next**. Leave blank all fields not included in the table. After you enter the values for a resource, configure it to use the subdeployment and click **Finish**.

Table 8. JMS Resources for a WebLogic Managed Server

Resource Type	Name	JNDI Name
Queue	SC scheduleTimerQueue	queue/schedulerTimerQueue
Queue	PM Validations Queue	queue/Validation
Queue	SC TaskProcessorQueue	queue/taskProcessorQueue
Queue	RF Executions Queue	queue/ReportingExecutions
Topic	SC taskStopperTopic	topic/taskStopperTopic

8 If the SOA Systinet host is a cluster create the resources listed in the Table 9.

Click **New** in the Resources table. Select the resource type and click **Next**. Leave blank all fields not included in the table. After you enter the values for a resource, target your cluster and click **Finish**.

Table 9. JMS Resources for a WebLogic Cluster

Resource Type	Name	JNDI Name
Distributed Queue	SC scheduleTimerQueue	queue/schedulerTimerQueue
Distributed Queue	PM Validations Queue	queue/Validation
Distributed Queue	SC TaskProcessorQueue	queue/taskProcessorQueue
Distributed Queue	RF Executions Queue	queue/ReportingExecutions
Distributed Topic	SC taskStopperTopic	topic/taskStopperTopic

9 Click **Activate Changes**.

Setting Startup Parameters

To operate SOA Systinet correctly, you must specify startup parameters.

To set the startup parameters:

1. In your browser, open the WebLogic Administration Console:

   ```
   http://localhost:7001/console
   ```

2. Log in as the WebLogic administrator.
3. Click **Lock & Edit**.
4. In the Domain Structure section, expand **Environment**, and select **Servers**.
5. Click the server created in Setting Up the WebLogic Server on page 76, and select the Server Start tab.
6. Set the class path to the following:

   - For WebLogic 9.2:
     `JAVA_HOME/lib/tools.jar;WL_HOME/server/lib/weblogic_sp.jar;WL_HOME/server/lib/weblogic.jar`
   - For WebLogic 10.0: `JAVA_HOME/lib/tools.jar;WL_HOME/server/lib/weblogic_sp.jar;`
     `WL_HOME/server/lib/weblogic.jar;BEA_HOME/modules/features/weblogic.server.modules_10.0.0.0.jar;`
     `BEA_HOME/modules/features/com.bea.cie.common-plugin.launch_2.1.0.0.jar`

   JAVA_HOME is the location of the JDK WebLogic uses and BEA_HOME is set during WebLogic installation.

   On UNIX systems, use colons instead of semi-colons as the file separator.

7. Add the following to Arguments:

   - For JRockit:

     ```
     -Xms1024m -Xmx1024m
     ```

- For Sun or HP JDK:

  ```
  -Xms1024m -Xmx1024m -XX:MaxPermSize=512m
  ```

  Memory sizing should take performance requirements into consideration for the deployed system. These settings are only a recommendation.

  If you use a UNIX operating system, also add the following property:

  Generic JVM Arguments `-Djava.awt.headless=true`

  If you use an SUSE Linux operating system, also add the following argument:

  ```
  -Djavax.xml.transform.TransformerFactory=net.sf.saxon.TransformerFactoryImpl
  ```

8 Click **Save**.

9 Click **Activate Changes**.

Setting Up WebSphere

WebSphere requires initial configuration before you can deploy SOA Systinet to it.

WebSphere setup is described in the following sections:

- Setting Startup Parameters on page 101
- Setting Up a WebSphere Cluster on page 102

Creating a Profile

Create a clean WebSphere profile with the *Cell* environment. This profile is stored in `WS_HOME/AppServer/profiles/PROFILE_NAME`. Referred to in this documentation as PROFILE_HOME.

Create application servers and/or clusters in the new profile as required by your deployment design. For details, see Designing Your Deployment on page 14.

If you are using a web server such as IBM HTTP Server (IHS) as a proxy or load balancer, register it with the Deployment Manager. For details, see the WebSphere Help.

Creating a Mail Session

Create a mail session using the WebSphere Administration Console.

To create a mail session:

1. Open the WebSphere Administration Console:

   `http://localhost:9060/ibm/console`

2. Select **Resources→Mail→Mail Sessions**.

3. Select your cell in the **Scope** drop-down field and click **New**.

4. Specify the mail session parameters as follows:

   - A unique, arbitrary descriptive name, for example `SOA Systinet Mail`
   - The JNDI name `/Mail`
   - Connection settings as per company email setup
   - SMTP credentials if required

Creating JDBC Resources

SOA Systinet requires an XA-enabled JDBC datasource to communicate with the database. JMS messaging requires a non-XA datasource.

Before creating these two datasources, you must create a JDBC provider for each of them.

SOA Systinet uses XA transactions. The application server transaction manager should be configured to have a minimum of 5 minutes for XA transaction timeout.

For details, see your application server documentation.

Open the WebSphere Administration Console (http://localhost:9060/ibm/console) and create JDBC resources, in the order of the following sections:

1. To create a JDBC provider for an XA datasource:
2. To create a JDBC provider for a non-XA datasource:
3. To create an XA-enabled JDBC data source:
4. To create a non-XA-enabled JDBC datasource (used by JMS):

To create a JDBC provider for an XA datasource:

1. Select **Resources→JDBC→JDBC Providers**.
2. For Scope, select your cell, and click **New**.
3. Select your database type.
4. Under Provider, select the driver for your database type.

For DB2, if there is more than one driver available, select the DB2 Universal driver.

For Oracle, if there is more than one driver available, select a "Thin" driver.

5 For the implementation type, select **XA data source**.

The **Name** is automatically completed with the driver name, followed by **(XA)**.

6 For the value of the variable `${driver_name_PATH}`, enter the location of the driver:

`ojdbc14.jar` (Oracle) or `db2jcc.jar` and `db2jcc_license_cu.jar` (DB2). The DB2 driver files are in `IBM_HOME/SQLLIB/java` by default.

7 Click **Finish**.

If you get the error "Connection not available, Timed out waiting for 180000" in the log file, you can do the following:

- Increase the Maximum connections property in **Resources→JDBC→Datasources→HP SOA Systinet DS for Oracle→Connection pool properties** (or specify 0 for no connection count limit)
- Increase the value Connection Timeout property in **Resources→JDBC→Datasources→HP SOA Systinet DS on Oracle→Connection pool properties** (default is 180 seconds). Setting 0 (infinite waiting for connection) is not recommended because of a risk of deadlocks.

To create a JDBC provider for a non-XA datasource:

- Repeat To create a JDBC provider for an XA datasource:, with the following exception:

  Select the implementation type **Connection pool data source**.

  The automatically generated name should not end in **(XA)**.

If you get the error DSRA3602E, refer to Wsadmin scripting fails with DSRA3602E [http://www-1.ibm.com/support/docview.wss?uid=swg21257580].

To create an XA-enabled JDBC data source:

1. Select **Resources→JDBC→Data Sources**.
2. For Scope, select your cell, and then click **New**.
3. Give the data source a unique, arbitrary descriptive name, for example, `HP SOA Systinet DS`.
4. Give the datasource the JNDI name `hpsoasystinetDS`.
5. Click **Next**.

   The Select JDBC Provider page opens.
6. Select **Select an existing JDBC provider**, select the XA JDBC provider you created in To create a JDBC provider for an XA datasource:, and then click **Next**.

   The Database Properties page opens.
7. The database properties you enter depend on the type of database you are using:

   For DB2:

   - Enter the database name, such as `platform`. Your database administrator can tell you this name.
   - From the **Driver type** drop-down field, select driver type "4."
   - Enter the server name.
   - Enter the port number if it differs from the default `50000`.
   - Deselect **Use this datasource for CMP**.

   For Oracle:

   - Type the full URL of the database you plan to use for SOA Systinet, such as `jdbc:oracle:thin:@server:1521:database`
   - From the drop-down field, select the data store helper class name for your version of the database.

- Deselect **Use this datasource for CMP**.

8 Click **Finish**.

9 Open the newly created data source and create an authentication alias:

- Click **JAAS - J2C Authentication Data**.

  A list of authentication aliases opens.

- Click **New**.
- Give an arbitrary string value for the alias, for example, HP SOA Systinet Credentials.
- For credentials, enter the user name and password for the database you use with SOA Systinet.
- Click **Finish**.

10 Reopen the newly created datasource and apply the new authentication alias:

- In Component-Managed Authentication Alias, select the authentication alias you created in Step 9.
- Under Authentication Alias for XA Recovery, select **Use Component-Manager Authentication Alias**.
- Under Container-Managed Authentication, for the Mapping Configuration Alias, select **DefaultPrincipalMapping**.
- Click **OK**.

11 Reopen the datasource and increase its connection pool size:

- Under Additional Properties, click **Connection pool properties**.

  The Connection Pool page opens.

- For Maximum Connections, type a number equal to at least 1/4 of the number of parallel requests that you require to be handled simultaneously. If you do not have an estimate of this number, set the maximum connections to 100.
- For Minimum Connections, type a number equal to the number of expected concurrent users.
- Click **OK**.

12 Click **Test connection** to make sure that your datasource configuration is correct.

To create a non-XA-enabled JDBC datasource (used by JMS):

- Repeat To create an XA-enabled JDBC data source:, with the following exceptions:
  - Give the non-XA datasource the JNDI name, `jms-hpsoasystinetDS`.
  - Select the non-XA JDBC provider.
  - Use the same authentication alias you created for the XA-enabled datasource.

After creating JDBC resources, restart the WebSphere Deployment Manager.

Creating a Messaging Bus

In WebSphere, JMS communication and the persistant storage of that communication are handled via a bus.

To create a messaging bus:

1 Open the WebSphere Administration Console:

```
http://localhost:9060/ibm/console
```

2 Select **Service Integration→Buses**.

The Buses page opens.

3 Click **New**.

The Create a New Messaging Engine Bus wizard opens.

4 Give the bus a unique, arbitrary descriptive name, for example, `SOABus`.

5 Deselect **Bus Security**, as it is not required, and click **Next**.

6 Click **Finish**.

The Buses page reopens.

7 Click the name of the bus you created to open its details page.

8 Click **Bus members**.

The Bus members page opens.

9 Click **Add**.

The Add a New Bus Member wizard opens.

10 Select a standalone server or cluster that will host SOA Systinet, and click **Next**.

The Select Type of Message Store page opens.

11 Select **Data store** for the type of message store, and click **Next**.

12 Enter the message store properties for the bus.

You can use the existing `jms-hpsoasystinetDS` data source but you might prefer to use a different data source for performance reasons.

If you use an existing data source, for the Schema Name, type the database user name, set the Authentication Alias to `HP SOA Systinet Credentials`, select **Create Tables**, and then click **Next**.

13 Review your selected options and click **Finish**.

The Bus Members page reopens.

14 Repeat Step 9 to Step 13 for every standalone server and cluster that will host SOA Systinet.

15 Return to the bus details page.

The Configuration tab is open by default.

16 Under Destination Resources, click **Destinations**.

A table of destinations opens.

17 Add the following destinations by clicking **New**:

Destination type	Identifier
Queue	scheduleTimerQueue
Queue	RFReportingExecutions
Queue	Validation
Queue	taskProcessorQueue
Topic Space	taskStopperTopic

Setting Up JMS

SOA Systinet requires JMS messaging resources that you must set up in the WebSphere Administration Console.

You can configure JMS to meet your requirements. This section describes a JMS setup that ensures that JMS resources are accessible by SOA Systinet and function correctly.

To set up JMS:

1 Open the WebSphere Administration Console:

```
http://localhost:9060/ibm/console
```

2 Select **Resources**→**JMS**.

3 Add the JMS resources listed in Table 10.

For each resource:

- Under JMS, click the resource type.
- Select the scope created in Creating a Profile on page 90, and click **New**.
- Select **Default Messaging Provider**, and click **Next**.
- Use the parameters from Table 10.

  Use the bus you created in Creating a Messaging Bus on page 95.

  Where a Queue Name, or Topic Space is required, select the relevant queue or topic that you created in Creating a Messaging Bus on page 95 Step 17.
- Click **OK**.

Table 10. JMS Resources

Resource Type	Name	JNDI Name
Queue Connection Factory	RF Connection Factory (Send)	jms/ReportingSenderConnectionFactory
Queue Connection Factory	RF Connection Factory (Rec)	jms/ReportingReceiverConnectionFactory
Queue Connection Factory	SOA Queue Connection Factory	jms/SOAQueueConnectionFactory
Topic Connection Factory	SOA Topic Connection Factory	jms/SOATopicConnectionFactory
Queue	SC scheduleTimer Queue	queue/scheduleTimerQueue
Queue	PM Validations Queue	queue/Validation
Queue	SC TaskProcessorQueue	queue/taskProcessorQueue
Queue	RF Executions Queue	queue/ReportingExecutions
Topic	SC taskStopperTopic	topic/taskStopperTopic

4 Modify the connection pool for the SOA Queue and the SOA Topic connection factories, by doing the following:

- Select the connection factory to modify.
- Under Additional Properties, click **Connection Pool Properties**.
- Change Maximum Connections to `100`.
- Click **OK**.

5 Select **Resources→Asynchronous Beans**, and select **Work Managers.**

6 Select the scope you created in Creating a Profile on page 90, and click **New** to create the following work managers:

- **SC Work Manager**

  Set JNDI Name `/wm/platform`, change Maximum Number of threads to `100`, and select **Growable**.

  The leading forward slash is required for the JNDI name in this case.

- **RF Work Manager**

  Set JNDI Name `wm/reporting`.

7 Set the JMS Activation Specifications listed in Table 11.

- Under JMS, click the **Activation Specifications**.
- Select the scope created in Creating a Profile on page 90, and then click **New**.
- Select **Default Messaging Provider**, and then click **Next**.
- Use the parameters from Table 11.

Use Destination Type `Queue`.

Use the bus you created in Creating a Messaging Bus on page 95.

- Click **OK**.

Table 11. JMS Resources

Name	JNDI Name	Destination JNDI Name
RF Activation	jms/RFActivation	queue/ReportingExecutions
PM Activation	jms/PMActivation	queue/Validation
PL Scheduler Timer Queue Activation	jms/PLSchedulerTimerQueueActivation	queue/scheduleTimerQueue
PL Task Runner Queue Activation	jms/PLTaskRunnerQueueActivation	queue/taskProcessorQueue
PL Task Stopper Topic Activation	jms/PLTaskStopperTopicActivation	topic/taskStopperTopic

8 Expand **Service Integration**, and select **Buses**.

9 Select the bus you created in Creating a Messaging Bus on page 95.

10 In the Topology section, click **Messaging Engines**.

11 Copy the name of the messaging engine to the clipboard.

12 Select **Resources→JMS**, and select **Activation Specifications**.

13 "Maximum Concurrent Endpoints" must be decreased to 4 on "PM Activation" in JMS - Activation specifications.

14 Select **PL Task Stopper Topic Activation**.

15 In the Subscription Durability section, add the following parameters:

- Leave Subscription Durability as non-durable.
- Enter a subscription name, for example, PL Task Stopper Subscription Name
- Enter a client identifier, for example, PL task subscription id
- Paste the messaging engine name as the Durable Subscription Home.

16 Click **OK**.

Setting Startup Parameters

SOA Systinet requires several parameters to be set in order to function correctly.

Use the WebSphere Administration Console to set these startup parameters.

To set the startup parameters:

1 In your browser, open the WebSphere Administration Console:

```
http://localhost:9060/ibm/console
```

2 Expand **Servers**, and select **Application Servers**.

3 Select the server.

4 In the Server Infrastructure section, expand **Java and Process Management**, and select **Process Definition**.

5 In the Additional Properties section, select **Java Virtual Machine**.

6 Set the following properties:

- Initial Heap Size `1000`
- Maximum Heap Size `1500`
- Generic JVM Arguments `-XX:MaxPermSize=150m`

- Memory sizing should take performance requirements into consideration for the deployed system. These settings are only a recommendation.

- If you use a UNIX operating system, also add the following property:

  Generic JVM Arguments `Djava.awt.headless=true`

- If you use an SUSE Linux operating system, also add the following property:

  Generic JVM Arguments
  `Djavax.xml.transform.TransformerFactory=net.sf.saxon.TransformerFactoryImpl`

7 Click **OK**.

- You must restart the server for these changes to take effect.

Setting Up a WebSphere Cluster

Clustered deployment of SOA Systinet is very similar to standalone deployment.

In all the following setup procedures, make sure to do the following:

- Whenever you select deployment scope, choose the cluster itself.
- When a restart is necessary, restart the whole cluster, including all servers joined to the cluster.
- When configuring a cluster, configure all servers within the cluster.
- When you deploy SOA Systinet, map modules to servers by selecting the cluster and an instance of *IBM HTTP Server*.

The following procedure describes how to set up a proxied load balanced cluster with two servers running on one node.

To create a load balanced cluster:

1 Install and start *IBM HTTP Server*.

2 Create a new WebSphere cell (deployment manager and application server) profile, by doing the following:

- Start the WebSphere Profile Management Tool.
- Select **Cell**, and click **Next**.
- Select **Advanced profile creation**.
- Select **Deploy the administrative console** (recommended), and **Deploy the default application**.
- Enter a unique Deployment Manager Profile Name (DMGR_NAME), Application Server Profile Name (APPSRV_NAME), and select a location for the new profile, and then click **Next**.
- Enter the Deployment Manager Node Name and the Application Server Node Name.

  These become the node containing the clustered servers.

  If necessary, correct the Host Name and Cell Name (CELL_NAME), and then click **Next**.
- Select **Enable administrative security**, enter the administrator credentials, and then click **Next**.
- If required, change the port values, and click **Next**.
- Deselect **Run the deployment manager process as a Windows service**, and click **Next**.
- Click **Next**, and then **Create**.
- Deselect **Launch the First Steps Console**, and click **Finish**.

3 Start the deployment manager, application server node, and the application server:

a Execute the command:

```
PROFILE_HOME/DMGR_HOME/bin/startManager
```

b Execute the command:

```
PROFILE_HOME/APPSRV_HOME/bin/startNode
```

If you want to run one or more nodes of the cluster on different machines, follow this procedure for each machine.

To apply a cluster to other machines:

1 Install IBM WebSphere®.

2 Start the WebSphere Profile Management Tool.

3 Click **Next**, and select **Custom Profile**.

4 Select **Advanced Profile Creation**.

5 Enter the Profile Name, for example, HPsoaClusterAppsrv2, and Profile Directory (PROFILE2_HOME).

6 Select **Make this profile default** and click **Next**.

7 Enter the Node Name (for example, HpsoaClusterNode2), and click **Next**.

8 Enter the Deployment Manager Hostname or IP Address, pointing to an existing deployment manager in a cell you want the new node to federate with.

Set the credentials to administer the new deployment manager, optionally change the deployment manager port, and then click **Next**.

9 If necessary, change the port values, and click **Next**.

10 Start the node with the following command:

PROFILE2_HOME/bin/startNode

The new node should appear in the deployment manager admin console nodes listing, **System Administration→Nodes**.

4 To create the cluster:

a In your browser, open the WebSphere Administration Console:

```
http://localhost:9060/admin
```

The port may vary depending on your settings.

b Select **Servers**, and select **Clusters**.

c Click **New**.

d Enter a cluster name (CLUSTER_NAME), select **Configure HTTP session memory-to-memory replication**, and then click **Next**.

e In the Select Basis for First Cluster Member, select **Create the member by converting an existing application server**, and click **Next**.

f Enter a new member name, for example `server2`, and click **Add Member**.

g Add servers as required.

If you require a different node, select it from **Select Node**.

h Click **Next**, and **Finish**, and then **Save**.

i During the resources setup for SOA Systinet, described in the previous sections, use the CELL_NAME as the target for the resource.

j During SOA Systinet deployment, select the CELL_NAME as the target for deployment.

5 To add a web server node to the cluster:

a In your browser, open the WebSphere Administration Console:

```
http://localhost:9060/admin
```

The port may vary depending on your settings.

b Select **Servers→Web Servers**, and click **New**.

c Enter a Server Name, for example, `IHS_NAME`, and click **Next** twice.

d Enter the Web Server Location, which should be the installation directory for *IBM HTTP Server* (IHS_HOME), and the a Plug-in Installation Location, usually, `IHS_HOME/Plugins`.

e Click **Next**, and **Finish**, and then **Save**.

6 For debug purposes, add an alias to the default virtual host to enable direct access to applications on all clustered servers, by doing the following:

a In your browser, open the WebSphere Administration Console:

```
http://localhost:9060/admin
```

The port may vary depending on your settings.

b Select **Environment**, and select **Virtual Hosts**.

c Click **default_host**, and then click **Host Aliases**.

d Click **New**.

e Enter the value for the second clustered server port, usually `9081`, click **OK**, and then click **Save**.

Repeat this step for as many servers as you require, in addition to adding their ports.

7 Regenerate the routing information for the web server.

Select **Servers**→**Web Servers**, and select `IHS_NAME`, and then select **Generate Plug-in** and **Propagate Plug-in**.

8 To start the cluster:

- In your browser, open the WebSphere Administration Console:

  `http://localhost:9060/admin`

  The port may vary depending on your settings.

- Select **Servers**, and then **Clusters**.
- Click your `CLUSTER_NAME`.
- Click **Start**.
- When the server starts, you can validate if the requests are load balanced across the cluster.

  In your browser, use the URL:

  `http://localhost/snoop`

  The Snoop servlet page should appear, displaying the port number.

  Every time you refresh the page, the port number should change.

  `http://localhost/soa/web` should display the SOA Systinet user interface.

LDAP Accounts Integration

When you install SOA Systinet, you can select to use an external LDAP server to retrieve information about users and groups.

The following sections describe how to integrate accounts from an LDAP server into SOA Systinet:

- Automatic Service Discovery on page 108

  A brief explanation of automatic service discovery and its implications.

- LDAP Service Properties on page 108

  A list of JNDI properties of the LDAP server that must be known to SOA Systinet.

Automatic Service Discovery

The automatic discovery of LDAP servers means you do not have to hardwire the URL and port of the LDAP server. Instead you can use `ldap:///o=JNDITutorial,dc=example,dc=com` as a URL, and the real URL is deduced from the distinguished name `o=JNDITutorial,dc=example,dc=com`.

Automatic discovery of the LDAP service using the URL's distinguished name is supported only in Java 2 SDK, versions 1.4.1 and later, so make sure that your Java version supports this.

LDAP Service Properties

To integrate external accounts during SOA Systinet installation, do one of the following:

- In the GUI Installer, select **LDAP** in the Account Provider page, and set your LDAP properties.

  For details, see Using the GUI Installer on page 19.

- Set the LDAP properties in the `configuration.properties` file during manual deployment.

  For details, see Configuring the User Store on page 115.

SOA Systinet integration with LDAP uses a JNDI interface to connect to LDAP servers.

For more information, about the JNDI API, see http://java.sun.com/products/jndi/tutorial/ldap/connect/create.-html and http://java.sun.com/j2se/1.5.0/docs/guide/jndi/jndi-dns.html#URL .

The following JNDI properties must be known to the server:

Property Name	**Property Description**	**API Link**
Naming Provider URL	URL of the LDAP service.	http://java.sun.com/j2se/1.5.0/docs/api/javax/naming/Context.html#PROVIDER_URL
Initial Naming Factory	Java class for the initial naming factory.	http://java.sun.com/j2se/1.5.0/docs/api/javax/naming/Context.html#INITIAL_CONTEXT_FACTORY
Security Principal	The name of the security principal for anonymous read access to the directory service.	http://java.sun.com/j2se/1.5.0/docs/api/javax/naming/Context.html#SECURITY_PRINCIPAL
Password	Password of security principal.	http://java.sun.com/j2se/1.5.0/docs/api/javax/naming/Context.html#SECURITY_CREDENTIALS
Security Protocol	Name of the security protocol. Default is "simple."	http://java.sun.com/j2se/1.5.0/docs/api/javax/naming/Context.html#SECURITY_PROTOCOL

4 Deploying SOA Systinet

The GUI Installation Wizard described in Chapter 2, Basic Installation, can be used for production environments. However, HP Software recommend using command-line installation for production deployment as the configuration options can be much more complicated.

Command line options are available with the installation command.

For details, see Installation Command Line Options on page 112.

Command-line deployment consists of the following steps:

1 Extract the installation archive.

For details, see Unpacking the Distribution on page 113.

2 Set the configuration properties for your deployment.

For details, see Configuring the Deployment on page 113.

3 Prepare any optional additions to your deployment.

For details, see the following sections:

- Preparing Extensions on page 124
- Preparing Updates on page 125

4 Finish the installation in one of the following ways:

- With decoupled database deployment.

  For details, see Decoupled Database Deployment on page 125.

- Including database deployment.

  For details, see Finishing Installation on page 127.

5 For applications servers other than JBoss, you must deploy the EAR file created by the installer.

  For details, see Deploying the EAR File on page 128.

Throughout this chapter, `SOA_HOME` refers to the SOA Systinet installation.

Installation Command Line Options

As an alternative to using the GUI installer, there are a number of command line options available:

The installation command is:

java -jar hp-soa-systinet-3.00-RC04.jar [OPTIONS]

The available option, with their definitions, are as follows:

- **-h, --help**

  Display the available options or list the available scenarios or steps in the console.

- **-x, --extract *PATH***

  Extract the installation archive to the specified location.

- **-i, --install-to *SOA_HOME***

  Install SOA Systinet in console mode to the specified location. Normally used in conjunction with **-u**.

- **-s, --save-config *FILE***

  Execute the GUI Installation, but save the configuration to the specified file instead of installing SOA Systinet.

- **-a, --dbadmin-mode**

  Run the installation in decoupled database mode.

  For details, see Decoupled Database Deployment on page 125.

- **-u, --use-config *FILE***

  Use the properties in the specified file to override the default or current configuration properties.

- **--passphrase *PASSPHRASE***

  If you want to use password encryption, specify the passphrase to use for encryption.

- **-d, --debug**

  Execute the installation in debug mode. All properties, SQL statements, and installation details are output to `SOA_HOME/log/install.log`.

Unpacking the Distribution

SOA Systinet is distributed as a JAR file. To perform manual configuration, you must extract the archive first.

To unpack the JAR file:

- Execute the following command:

  java -jar hp-soa-systinet-3.00.jar -x SOA_HOME

Configuring the Deployment

The distribution contains a configuration properties file. You can configure all your deployment options in this file prior to deployment.

To edit the configuration file:

1. Open `SOA_HOME/conf/setup/configuration.properties` with a text editor.

2 In the configuration file, make the changes you need for your deployment.

For configuration details, see the following sections:

- Configuring the License on page 114
- Configuring the User Store on page 115
- Configuring the Administrator on page 117
- Configuring the Database on page 118
- Configuring the Endpoint on page 119
- Configuring an Application Server on page 121
- Configuring Password Encryption on page 122
- Configuring Custom Data Import on page 123
- Configuring an SMTP Server on page 123
- Configuring Full Text Search on page 124

3 Save your changes.

Configuring the License

If you have a license from your HP sales representative then you can use it in the configuration file.

To configure the license:

1 Open `SOA_HOME/conf/setup/configuration.properties` with a text editor.

2 Set `shared.license.licensed.to` to the license owner.

3 Set `shared.license.key` to your license key.

4 Save your changes.

Configuring the User Store

In the configuration, you can set the user store to be in the database or an LDAP server.

To configure the user store:

1. Open `SOA_HOME/conf/setup/configuration.properties` with a text editor.
2. Do one of the following:
   - Set `shared.um.account.backend.type=database` to use the database.
   - Set `shared.um.account.backend.type=ldap` to use an LDAP server.

It may be more convenient to integrate LDAP accounts after installation using the Setup Tool.

For details, see "Setting Up LDAP Integration" in the *HP SOA Systinet Administration Guide* .

To configure LDAP integration:

1. Open `SOA_HOME/conf/setup/configuration.properties` with a text editor.
2. Set the following properties according to your LDAP configuration:
   - `shared.um.java.naming.provider.url`
   - `shared.um.java.factory.initial=com.sun.jndi.ldap.LdapCtxFactory`
   - `shared.um.java.naming.security.principal`
   - `shared.um.java.naming.security.credentials`
   - `shared.um.java.naming.security.authentication=simple`
3. To enable LDAP account integration, set the following properties:

   Mandatory account properties:

- shared.um.account.backend.type=ldap
- shared.um.account.backend.enableMoreBackends=true
- shared.um.account.ldapLoginName
- shared.um.account.ldapFullName
- shared.um.uddi.ldap.searchfilter.user
- shared.um.uddi.ldap.searchbase.user

Optional account properties:

- shared.um.uddi.ldap.searchscope.user=2
- shared.um.uddi.ldap.searchMaxResults.user=100
- shared.um.account.ldapEmail
- shared.um.account.ldapDescription
- shared.um.account.ldapLanguageCode
- shared.um.account.ldapBusinessName
- shared.um.account.ldapPhone
- shared.um.account.ldapAlternatePhone
- shared.um.account.ldapAddress
- shared.um.account.ldapCity
- shared.um.account.ldapCountry
- shared.um.account.ldapBlocked
- shared.um.account.ldapZip

4 To enable LDAP group integration, set the following properties:

Mandatory group properties:

- `shared.um.group.backend.type=ldap`
- `shared.um.group.backend.enableMoreBackends=true`
- `shared.um.account.ldapMember`
- `shared.um.account.ldapName`

Optional group properties:

- `shared.um.uddi.ldap.searchfilter.group=2`
- `shared.um.uddi.ldap.searchbase.group=100`
- `shared.um.group.ldapDescription`
- `shared.um.account.ldapOwner`

5 Save your changes.

Configuring the Administrator

SOA Systinet requires an initial administrator user.

To configure the administrator:

1 Open `SOA_HOME/conf/setup/configuration.properties` with a text editor.

2 Set `shared.administrator.username` and `shared.administrator.password` to the credentials for your administrator.

3 Set `shared.notification.default.mail.from` to the mail address to be the source of admin messages.

4 Save your changes.

Configuring the Database

SOA Systinet must connect to a correctly configured database.

For details about setting up a database, see Preparing Databases on page 35.

To configure the database:

1. Open `SOA_HOME/conf/setup/configuration.properties` with a text editor.
2. Set `shared.db.driver.path` to the location of your JDBC driver.
3. Set `shared.db.type` to one of the following options:
   - db2
   - mssql
   - oracle
4. Set `shared.db.username` and `shared.db.password` to the values provided by your database administrator.
5. Set `shared.db.host`, `shared.db.port`, and `shared.db.dbname` to the values for your database.

   Alternatively, set `shared.db.url` to the connection string for your database.
6. Set `shared.db.action` to one of the following options:
   - To create a new schema in an existing database: `createSchema`
   - To create a new database/tablespace: `create`

   For more details, see Database Installation Types on page 36.
7. If you are creating a tablespace, set `shared.db.datafile` to the location of the tablespace, and set `shared.db.admin.username` and `shared.db.admin.password` to the database administrator credentials.
8. If you are using DB2, set `shared.db.tablespace`and `shared.db.pool` to the values set in Setting Up DB2 on page 38.

9 Save your changes.

Configuring the Endpoint

SOA Systinet is visible to the user as a web application and requires an endpoint.

You can configure the endpoint for a number of scenarios which should meet any connection requirement.

The default port numbers for each application server are different. Ensure that the port numbers you select match those required by your application server.

To configure the endpoint:

1 Open `SOA_HOME/conf/setup/configuration.properties` with a text editor.

2 Do one of the following:

- For HTTP only communication:

  Set `shared.hostname` to the required server, for example, `localhost`.

  Set `shared.http.port` to the required port, for example, `8080`.

  Do not set `shared.https.port`. No port is required for this scenario.

  Set `shared.https.use=false`.

  Set `shared.application.context` to your required deployment context, for example, `soa`.

- For HTTPS only communication:

  Set `shared.hostname` to the required server, for example `localhost`.

  Do not set `shared.http.port`. No port is required for this scenario.

  Set `shared.https.port` to the required secure port, for example, `8443`.

  Set `shared.https.use=false`.

Set `shared.application.context` to your required deployment context, for example, `soa`.

- For HTTP and HTTPS with enforcing secured URLs:

  Set `shared.hostname` to the required server, for example, `localhost`.

  Set `shared.http.port` to the required port, for example, `8080`.

  Set `shared.https.port` to the required secure port, for example, `8443`.

  Set `shared.https.use=true`.

  Set `shared.application.context` to your required deployment context., for example, `soa`.

- For HTTP and HTTPS without enforcing secured URLs:

  Set `shared.hostname` to the required server, for example, `localhost`.

  Set `shared.http.port` to the required port, for example, `8080`.

  Set `shared.https.port` to the required secure port, for example, `8443`.

  Set `shared.https.use=false`.

  Set `shared.application.context` to your required deployment context, for example, `soa`.

- For an HTTP only proxied endpoint:

  Set `shared.hostname` to the required proxy server, for example, `proxylhost`.

  Set `shared.http.port` to the required port, for example, `8080`.

  Do not set `shared.https.port`. No port is required for this scenario.

  Set `shared.https.use=false`.

  Set `shared.application.context` to your required deployment context, for example, `soa`.

- For an HTTPS only proxied endpoint:

  Set `shared.hostname` to the required proxy server, for example, `proxyhost`.

Do not set `shared.http.port`. No port is required for this scenario.

Set `shared.https.port` to the required secure port, for example, `8443`.

Set `shared.https.use=false`.

Set `shared.application.context` to your required deployment context, for example, `soa`.

- For HTTP and HTTPS proxied endpoints with enforcing secured URLs:

  Set `shared.hostname` to the required proxy server, for example, `proxyhost`.

  Set `shared.http.port` to the required port, for example, `8080`.

  Set `shared.https.port` to the required secure port, for example, `8443`.

  Set `shared.https.use=true`.

  Set `shared.application.context` to your required deployment context, for example, `soa`.

3 If your HTTPS communication is validated against SSL certificates, set `shared.ui.link.authentication=true`.

  For details about SSL Certificates, see SSL Certificates on page 152.

4 Save your changes.

Configuring an Application Server

You must deploy SOA Systinet to an application server which is correctly set up.

For details about setting up an application server, Setting Up Application Servers on page 46.

To configure an application server:

1 Open `SOA_HOME/conf/setup/configuration.properties` with a text editor.

2 To set the application server settings:

For JBoss:

- Set `shared.as.server=jboss`
- Set `shared.as.jboss.configuration` to the required configuration name:
- By default, this is `default`.
- Set `shared.as.jboss.location` to the location of your application server installation.

For Oracle Application Server:

- Set `shared.as.server=oas`

For WebLogic:

- Set `shared.as.server=weblogic`

For WebSphere:

- Set `shared.as.server=websphere`

3 Save your changes.

Configuring Password Encryption

By default, the installer does not encrypt passwords in the configuration file.

If you require passwords to be encrypted for communication with other servers, such as the database and SMTP, you can enable encryption.

To enable password encryption:

1 Open `SOA_HOME/conf/setup/configuration.properties` with a text editor.

2 Set `password.encryption.enabled=true`.

3 Save your changes.

When you continue the installation, you must apply a passphrase to encrypt any passwords in the configuration file.

For details, see Finishing Installation on page 127.

Configuring Custom Data Import

By default, the installer imports a default data set. You can configure the installation to import a custom data set instead.

The data set must conform to the requirements of the import tool.

For details, see "Import Tool" in the *HP SOA Systinet Administration Guide* .

To configure custom data import:

1. Open `SOA_HOME/conf/setup/configuration.properties` with a text editor.
2. Set `import.type=custom`.
3. Set `import.path` to the location of your data image.
4. Save your changes.

Configuring an SMTP Server

By default, the installer does not configure an SMTP server for mail notifications.

To configure an SMTP server:

1. Open `SOA_HOME/conf/setup/configuration.properties` with a text editor.
2. Set `platform.smtp.auth=true`
3. Set `platform.smtp.host` and `platform.smtp.host` to the location of your SMTP server.
4. Set `platform.smtp.auth.user` and `platform.smtp.auth.password` to an authorised SMTP server user.

5 Save your changes.

Configuring Full Text Search

By default, full text search is disabled in the installer.

To enable full text search:

1 Open `SOA_HOME/conf/setup/configuration.properties` with a text editor.

2 Set `shared.db.fulltextsearch=true`.

3 To disable the automatic addition of % to search terms, set `shared.db.fulltextsearch.appendpercentage=false`.

4 Save your changes.

After installation you must enable full text search for your database, and in the SOA Systinet UI.

- Configure your database for full text search.

  For details, see Enabling Full Text Search on page 146.

- Enable full text search in the SOA Systinet UI.

  For details, see "Configuration Options" in the *HP SOA Systinet Administration Guide* .

Preparing Extensions

Most production deployments involve a customization layer with organization specific modifications to the SOA Definition Model (SDM), UI customization, and customized reports. These extensions can be added to the installation directory prior to deployment.

To add extensions prior to installation:

1 Extract the distribution file.

  For details, see Unpacking the Distribution on page 113

2 Copy the extensions to the `SOA_HOME/extensions` folder.

Preparing Updates

To add updates prior to installation:

1 Extract the distribution file.

For details, see Unpacking the Distribution on page 113.

2 Copy the required updates to the `SOA_HOME/updates` folder.

Decoupled Database Deployment

In some production deployments. the database administrator may perform the creation of the tablespace and schema as a separate installation step.

For more details, see Manual Database Arrangement on page 37.

The installer enables you to separate the database deployment by creating scripts for the database administrator to run.

To perform decoupled database installation:

1 Extract the installation.

For details, see Unpacking the Distribution on page 113.

2 Configure the installation properties.

For details, see Configuring the Deployment on page 113.

For database details, see Configuring the Database on page 118.

In this scenario you only require the database type, `shared.db.type`, and the required action, `shared.db.action`, for the database properties.

3 Execute the following command:

SOA_HOME/bin/setup -c -a

Add **--passphrase *PASSPHRASE*** if you want to use password encryption.

The installation stops after the creation of scripts, to create the tablespace and schema, depending on the configuration properties.

4 Provide the scripts created by the installer to the database administrator.

The installer creates the scripts in `SOA_HOME/sql`.

- If you are creating a new database/tablespace execute `createdb.sql` and then execute `all.sql` to create the schema.
- To create the schema in an existing database, execute `all.sql`.
- The remaining scripts are required to create the schema and are called by `all.sql`.

The schema creation scripts contain drop instructions which can, by design, fail and their failure must be ignored. If you are overwriting an existing SOA Systinet 3.00 database, make sure that the SQL tool ignores these failures.

The database administrator must execute the scripts and create a *common user* to provide SOA Systinet users access to the database.

For details, see the relevant common user procedure for each database in Preparing Databases on page 35.

5 After the database administrator executes the scripts, set the database connection parameters in `configuration.properties` as advised by the database administrator.

For details, see Configuring the Database on page 118.

6 Execute the following command to finish the installation:

SOA_HOME/bin/setup -c

Add **--passphrase *PASSPHRASE*** if you want to use password encryption.

For more details about **setup -c**, see Finishing Installation on page 127.

Finishing Installation

The installation command, **java -jar hp-soa-systinet-3.00.jar**, can only be executed once as the installation directory must be empty.

Command-line installation requires you to extract the installation archive first, using the **-x** switch on the installation command.

This extracts the file structure and the `configuration.properties` file, enabling you to configure and prepare your deployment.

The Setup Tool enables you to continue installation after extraction.

To finish installation and deployment, do one of the following:

- Execute the command:

  SOA_HOME/bin/setup -c

- If you enabled password encryption in Configuring Password Encryption on page 122, execute the command:

  SOA_HOME/bin/setup -c --passphrase *PASSPHRASE*

  The installer uses *PASSPHRASE* to encrypt the password in `configuration.properties`.

  If you use a command line tool that requires authentication in another server, you must add the option **--passphrase *PASSPHRASE*** to the command.

The Setup Tool is a GUI utility which enables administrators to reconfigure a deployment of SOA Systinet. The **-c** switch runs the Setup Tool in console mode, continuing the installation from the point at which it previously stopped.

For details about the Setup Tool, see "Setup Tool" in the *HP SOA Systinet Administration Guide* .

Deploying the EAR File

For non-JBoss application servers, you must deploy the EAR file created by the installer using application server functionality.

Additionally, you may need to modify the EAR file to apply authentication requirements.

If you are redeploying an EAR file, do not use the redeploy functionality of your application server. Undeploy the EAR file and then deploy the EAR file as described in this section.

If you are deploying to a cluster, you can only have one cluster server running. After a successful deployment, the deployment is copied to other cluster servers when you start them.

The SOA Systinet EAR file is updated during installation or by the application of extensions and updates. It contains JSPs that are compiled by the application server during deployment. The compilation of JSPs may take some time to complete.

HP Software recommend that you precompile JSPs before deployment, specifically for Oracle Application Server and WebLogic.

Use the following script to create an EAR file with precompiled JSPs.

```
SOA_HOME/deploy/AS/jspc/precompile
```

AS is an application server specific folder name (jboss, oas, wl, or ws). The script may require some environment variables to be set. If they are not set, the script fails and outputs the name of the required environment variable.

The script creates `SOA_HOME/deploy/precompiled.ear` which can be used instead of `SOA_HOME/deploy/hp-soa-systinet.ear` during deployment.

For details, see the following sections:

Setting-up Authentication

By default, SOA Systinet requires authentication for selected web resources. The configuration of these requirements conforms to the J2EE specification, as part of the deployment descriptors contained in the SOA Systinet EAR file.

Table 12 describes the default authentication method with the URL patterns, relative to the deployment context of the EAR file (the default is `soa`).

Table 12. Authentication Methods

Authentication Method	URL Patterns
Form Authentication (required by the web UI)	web/service-catalog/* (Service Catalog UI)
	web/policy-manager/* (Policy Manager UI)
	web/shared/* (shared UI)
Basic Authentication (HTTP) (required by parts of the REST interface and self-tester)	systinet/platform/restBasic/* (see "Proprietary REST Interface" in the *HP SOA Systinet Developer Guide*)
	platform/restSecure/* (see "Atom-Based REST Interface" in the *HP SOA Systinet Developer Guide*)
	policymgr/restSecure/* (Policy Manager REST interface)
	reporting/restSecure/* (Reporting REST interface)
	self-test/secure-snoop (see SOA Systinet Self-Tester on page 160)
No authentication	web/design/* (static UI resources such as images)
	systinet/platform/rest/* (see "Proprietary REST Interface" in the *HP SOA Systinet Developer Guide*)
	platform/rest/* (see "Atom-Based REST Interface" in the *HP SOA Systinet Developer Guide*)
	policymgr/rest/* (Policy Manager REST interface)
	reporting/rest/* (Reporting REST interface)
	self-test (excluding secure-snoop page, see SOA Systinet Self-Tester on page 160)

The SOA Systinet EAR contains various WAR files. Some of the presented web pages may include links between resources contained in different WAR files. The security context (knowledge of the authenticated user) may be lost when following such links, so you may be prompted to sign in again.

Application servers provide a single-sign-on (SSO) solution for this situation:

- **JBoss**

  SSO is set up during SOA Systinet installation.

For details, see http://www.jboss.org/wiki/Wiki.jsp?page=SingleSignOn .

- **Oracle Application Server**

  SSO must be enabled for the newly developed application.

  For details, see http://download.oracle.com/docs/cd/B31017_01/web.1013/b28957/javasso.-htm#BABIDDAC.

- **WebLogic**

  SSO is already set up in the deployment descriptor in the deployed EAR file.

- **WebSphere**

  The SSO option is switched on when you enable administrative security.

Role Mapping

SOA Systinet requires one J2EE role, `authenticated`. By default, this role is mapped to any authenticated user for all application servers. If required, you can change the mapping of this role to grant or deny access for selected users that pass authentication.

For details, see the relevant security documentation for your application server.

SOA Systinet also contains an `administrator` role, which enables privileged access to all SOA Systinet resources independently of ACLs, as well as access to SOA Systinet administration tasks.

This role is managed by SOA Systinet and not by the application server. The initial administrator name is set during installation of SOA Systinet. Any administrator can use the SOA Systinet UI to assign the administrator role to additional users or user groups.

Deploying the EAR to OAS

The SOA Systinet installer does not deploy the EAR file to *Oracle Application Server* (OAS), you must deploy it using OAS functionality.

If you are *re-deploying* an SOA Systinet EAR to OAS, *undeploy* the existing application and restart OC4J before deploying the updated EAR file.

To deploy the SOA Systinet EAR to OAS:

1 Setup your OAS server, specifically JDBC and JMS resources.

For details, see Setting Up Oracle Application Server on page 64.

2 Do one of the following:

- Run command-line installation using OAS parameters.

  For details, see Chapter 4, Deploying SOA Systinet.

- Run the GUI installation using OAS parameters.

  For details, see Chapter 2, Basic Installation.

3 Start OAS with the command:

OAS_HOME/opmn/bin/opmnctl startall

4 Open the Application Server Control:

`http://localhost/em`.

5 Log in as the OC4J administrator.

6 Click the link to your OC4J instance (by default named Home) in the tree view on the main page.

7 Open the Applications tab.

8 Click **Deploy**.

The deployment wizard opens.

9 Do one of the following:

- Select **Archive is present on local host**, and **Browse** or enter the path to the EAR file.
- Select **Archive is already present**, and enter the path to the EAR file.

The default location of the EAR file is `SOA_HOME/deploy/hp-soa-systinet.ear`.

10 Keep the option **Automatically create a new deployment plan**, and click **Next**.

11 Enter the application name, `HP SOA Systinet`.

12 Leave all other fields at default, and click **Next**.

A page with the deployment details opens.

13 SOA Systinet relies on OAS to perform authentication. Your company policy determines the choice of a security provider for authentication.

The default security provider is usually file-based. Users and groups in a default file-based provider can be created using the OAS admin functions, but changes to users or groups may require an OC4J server restart to become effective.

You can also select an LDAP-based security provider.

For more details, see the *Oracle Containers for J2EE Security Guide*.

If you need to change the default security provider:

- In the Select Security Provider row, click **Go to Task**.
- Select the security provider you want. For example, for integration with Sun ONE Directory Server or Active Directory, select **Third Party LDAP Server**.
- Specify the provider details, and click **OK**.

14 Review the deployment details, and if they are correct click **Deploy**.

Deployment may take several minutes.

15 To verify successful deployment, in your browser, view:

`http://hostname:port/context/self-test`

A report page, HP SOA Systinet Self-test, is displayed and there should be no errors.

For self-test details, see SOA Systinet Self-Tester on page 160.

16 If you use a default file based security provider, you can manage users using the Oracle Application Server Control:

a Open your OC4J instance.

b Select the **Administration** tab.

c Open **Security→Security Providers**.

d Click **Instance Level Security**.

e Select the **Realms** tab.

f Use **Create** to create the users you require including the admin user specified during installation.

Roles are not required.

Deploying the EAR to WebLogic

The SOA Systinet installer does not deploy the EAR file to WebLogic, you must deploy it using WebLogic functionality.

If you use an SUSE Linux operating system, copy the following file:

`SOA_HOME/lib/saxon-8.*.jar` to `DOMAIN_HOME/lib`

To deploy the SOA Systinet EAR to WebLogic:

1 Setup your WebLogic server.

For details, see Setting Up WebLogic on page 75.

2 Do one of the following:

- Run command-line installation using WebLogic parameters.

  For details, see Chapter 4, Deploying SOA Systinet.

- Run the GUI installation using WebLogic parameters.

  For details, see Chapter 2, Basic Installation.

3 Start the WebLogic server.

4 In your browser, open the WebLogic Administration Console:

`http://localhost:7001/console`

5 Log in with the administrator credentials created in Creating Domains for SOA Systinet on page 75.

6 In the console, click **Lock & Edit**.

7 In the Domain Structure section, select **Deployments**, and click **Install**.

8 Navigate to `SOA_HOME/deploy/`, select `hp-soa-systinet.ear`, and then click **Next**.

9 Select **Install this deployment as an application**, and click **Next**.

10 Select the managed server or cluster you want to host SOA Systinet, and click **Next**.

11 In the Security section, select **DD Only**, and click **Finish**.

12 Click **Activate Changes**.

Start SOA Systinet using the **Start** on the Deployments page.

To verify that the SOA Systinet deployment is running, view self-test in a browser window, at `http://hostname:port/context/self-test`.

Deploying the EAR to WebSphere

The SOA Systinet installer does not deploy the EAR file to WebSphere, you must deploy it using WebSphere functionality.

If you use an SUSE Linux operating system, copy the following file:

`SOA_HOME/lib/saxon-8.*.jar` to `PROFILE_HOME/lib`

To deploy the SOA Systinet EAR to WebSphere:

1 Setup your WebSphere server.

For details, see Setting Up WebSphere on page 89.

2 Do one of the following:

- Run command-line installation using WebSphere parameters.

  For details, see Chapter 4, Deploying SOA Systinet.

- Run the GUI installation using WebSphere parameters.

  For details, see Chapter 2, Basic Installation.

3 In your browser, open the Administration Console:

`http://localhost:9060/ibm/console`

4 Expand **Applications**, and select **Enterprise Applications**.

5 Click **Install**.

6 Click **Browse**, and navigate to `SOA_HOME/deploy/`, and then select `hp-soa-systinet.ear`.

7 Select **Prompt me only when additional information is required**, and click **Next**.

8 Set the following options:

- Precompile JavaServer Pages files
- Distribute application
- Allow dispatching includes to remote resources
- Allow servicing includes from remote resources

9 Map modules to servers by selecting the servers to deploy SOA Systinet.

10 Map modules to servers by selecting a module and virtual host.

11 Proceed to the Summary step, and click **Finish**.

12 Wait for the deployment to finish, and click **Save**.

13 Expand **Applications**, and select **Enterprise Applications**.

14 Select **HP SOA Systinet**.

15 In the Detail Properties section, click **Class loading and update detection**.

16 Set the following properties:

- Polling interval `0`
- Classes loaded with application class loader first
- Single class loader for application

17 Click **OK**, and save your changes.

18 Expand **Security**, and select **Secure administration, applications, and infrastructure**.

19 Select **Enable application security**.

20 In the Authentication section, expand **Web Security**, and select **General Settings**

21 Select **Use available authentication data when an unprotected URI is accessed** and click **OK**.

22 In the Configuration page, click **Apply**.

23 You can set users and roles if required.

To create a user:

1 Expand **Users and Groups**, and select **Manage Users**.

2 Click **Create**.

3 Enter the user parameters, and click **Create**.

24 Expand **Applications**, and select **Enterprise Applications**.

25 Select the check-box for **HP SOA Systinet**, and click **Start**.

➤ SOA Systinet starts automatically, whenever the server is started.

➤ The HP SOA Systinet log can be viewed in the file:

`PROFILE_HOME/logs/server_name/trace.log`.

5 After Installation

SOA Systinet may require some configuration after installation.

If you are upgrading from SOA Systinet 2.52 you can migrate your data after installing SOA Systinet 3.00.

For details, see "Data Migration" in the *HP SOA Systinet Administration Guide* .

This chapter contains the following sections:

Launching SOA Systinet on JBoss

The `SOA_HOME/bin` directory of SOA Systinet contains the scripts `serverstart`, `serverstop` and `env-jboss`.

Running `serverstart` calls `env-jboss`, which sets environment parameters for JBoss. Specifically, `env-jboss` gives JBoss access to the SOA Systinet client truststore and optimizes JBoss memory allocation.

In some production environments, where the SOA Systinet is widely distributed or clustered, or when there are applications other than SOA Systinet on the same JBoss server, or where JBoss is using non-default configuration or rmi ports, it might be preferable to use the native JBoss `run` scripts.

In this case, you must first modify the `run` script of each host JBoss as described in Modifying the JBoss Run Script on page 55. The contents of `serverstart` and `serverstop` are also useful guides in this case.

Setting up the User Store in JBoss

By default, SOA Systinet uses a JBoss user store to authenticate users. The default user store is a plain text file `JBOSS_PROFILE/conf/users.properties`, which contains lines with `USERNAME=PASSWORD`. All users listed in this file can authenticate with SOA Systinet.

SOA Systinet defines a new JBoss security domain that you can customize to setup authentication against various user stores, including LDAP. The definition of this domain is contained in `SOA_HOME/deplot/jboss/hp-soa-systinet.sar`, which is deployed to JBoss during installation.

To modify the JBoss authentication:

1. Extract `SOA_HOME/deploy/jboss/hp-soa-systinet.sar` to a directory.
2. In the unzipped directory, open `hp-systinet-login-config.xml` with a text editor.
3. Change the login module definitions as required.

   For details, see the JBoss security documentation.

   Example 1 on page 141 is an excerpt of the relevant section of this file.
4. Zip the directory back to `hp-soa-systinet.sar`.
5. Redeploy the SAR file to `JBOSS_PROFILE/deploy/hp-soa-systinet.sar` and restart JBoss.

Example 1: SOA Systinet JBoss Login Configuration File

```
<!DOCTYPE policy PUBLIC "-//JBoss//DTD JBOSS Security Config 3.0//EN"
"http://www.jboss.org/j2ee/dtd/security_config.dtd">
<policy>
  <application-policy name="hp-systinet">
    <authentication>
      <!-- ================================== -->
      <!-- CLIENT CERT authentication EXAMPLE -->
      <!-- ================================== -->
      <!-- JBOSS's SSL client certificate mapping, uncomment when using
        CLIENT-CERT login method -->
      <!--
        <login-module
        code="org.jboss.security.auth.spi.BaseCertLoginModule" flag="required">
        <module-option name="password-stacking">useFirstPass</module-option>
        <module-option
        name="verifier">org.jboss.security.auth.certs.AnyCertVerifier</module-option>
        <module-option
        name="securityDomain">java:/jaas/hp-systinet</module-option>
        </login-module>
      -->
      <!-- ====================================== -->
      <!-- USERNAME/PASSWORD authentication EXAMPLE -->
      <!-- ====================================== -->
      <!-- JBOSS's login module that verifies name and password against users.properties -->
      <!-- file from the classpath (classpath contains JBOSS's configuration conf directory -->
      <login-module code="org.jboss.security.auth.spi.UsersLoginModule"
        flag="required">
        <module-option name="password-stacking">useFirstPass</module-option>
      </login-module>
      <!-- JBOSS's login module that verifies name and password against LDAP -->
      <!--
        <login-module code="org.jboss.security.auth.spi.LdapLoginModule"
        flag="required">
        <module-option
        name="java.naming.factory.initial">com.sun.jndi.ldap.LdapCtxFactory</module-option>
        <module-option
        name="java.naming.provider.url">ldap://localhost:63284</module-option>
        <module-option
        name="java.naming.security.authentication">simple</module-option>
        <module-option name="principalDNPrefix">uid=</module-option>
        <module-option
        name="principalDNSuffix">,ou=people,dc=your,dc=company</module-option>
        </login-module>
```

```
      -->
      <!-- ======================================== -->
      <!-- Mandatory Role Mapping, authenticated users -->
      <!-- will become members of "authenticated" role -->
      <!-- ======================================== -->
      <!-- custom login module is used to assign authenticated role -->
      <login-module
        code="com.hp.systinet.security.jboss.AssignRoleLoginModule" flag="required">
        <module-option name="role">authenticated</module-option>
      </login-module>
    </authentication>
  </application-policy>
</policy>
```

Logging

SOA Systinet uses log4j logging, a popular logging package for Java that offers various options to change the logging format and/or output.

This section describes the default use of log4j logging by SOA Systinet.

You can change the logging options to suit your needs.

For more details, see the log4j manual at http://logging.apache.org/log4j/1.2/manual.html.

Log4j Configuration

SOA Systinet relies on the log4j configuration chosen using the "Default Initialization Procedure", described in http://logging.apache.org/log4j/1.2/manual.html.

This default initialization procedure results in the following configuration:

- The default logging configuration, as detailed in Example 2 on page 144, is used for the SOA Systinet EAR file deployed to OAS, WebLogic, WebSphere. The file `log4j.properties`, which is contained in the EAR file, contains the default configuration.
- The option, *`-Dlog4j.configuration=file:/ABSOLUTE_LOG4J_CONFIG_FILE_PATH`*, can be set for the execution of the SOA Systinet server start command. This option forces log4j to use the specified configuration.

SOA Systinet tools execute a java command with a *-Dlog4j.configuration* option that points to a SOA_HOME/conf/log4j.config.

SOA Systinet creates log files for these tool executions in the SOA_HOME/log directory.

The option, *-Dlog4j.configuration=file:/file location*, can be added to the command that starts your application server. This enables you to override the default configuration contained in the EAR file.

- The logging configuration for an EAR deployed to JBoss is updated during installation, the content of this configuration is similar to the default properties, but is expressed in an XML file.

  - JBOSS_HOME/server/JBOSS_PROFILE/conf/jboss-log4j.xml

  The audit log file is created in the JBOSS_HOME/server/JBOSS_PROFILE/log directory. The Application log is a part of the default JBoss log output (the console and also the JBOSS_HOME/server/JBOSS_PROFILE/log/server.log file).

If you are not sure about the logging configuration, do one of the following:

- Use the SOA Systinet Self-Tester, which reports the location of the log4j configuration in use.

  For details, see SOA Systinet Self-Tester on page 160.

- Add the option *-Dlog4j.debug* to the application server start command and restart the application server.

  Log4j then outputs configuration messages to the console.

Default Log4j Configuration

The default log4j configuration from a deployed SOA Systinet is shown in Example 2 on page 144.

SOA Systinet tools use the configuration from SOA_HOME/conf/log4j.config, which may be different.

Example 2: Log4j Configuration File

```
# put all logs to console and a log file
log4j.rootLogger=INFO,stdout,file

# console appender
log4j.appender.stdout=org.apache.log4j.ConsoleAppender
log4j.appender.stdout.Target=System.out
log4j.appender.stdout.layout=org.apache.log4j.PatternLayout
log4j.appender.stdout.layout.ConversionPattern=%p: %c{2} - %m%n

# file appender
log4j.appender.file=org.apache.log4j.RollingFileAppender
log4j.appender.file.maxFileSize=20MB
log4j.appender.file.maxBackupIndex=5
log4j.appender.file.File=log4j.log
log4j.appender.file.threshold=INFO
log4j.appender.file.layout=org.apache.log4j.PatternLayout
log4j.appender.file.layout.ConversionPattern=%d{ABSOLUTE} %5p %c - %m%n

# audit log appender
log4j.appender.Systinet_AUDIT=org.apache.log4j.RollingFileAppender
log4j.appender.Systinet_AUDIT.File=hpsoa_audit.log
log4j.appender.Systinet_AUDIT.MaxFileSize=10000KB
log4j.appender.Systinet_AUDIT.MaxBackupIndex=10
log4j.appender.Systinet_AUDIT.layout=org.apache.log4j.PatternLayout
# see http://logging.apache.org/log4j/1.2/apidocs/org/apache/log4j/PatternLayout.html
# for formating rules, following extra arguments can be moreover used to
# customize the format
#  %X{audit.eventId} - event ID
#  %X{audit.result} - event result
#  %X{audit.category} - event category
#  %X{audit.ctxId} - event context id
#  %X{audit.actor} - event actor
#  %X{audit.resource} - event actor
#  %X{audit.detail} - event detail
log4j.appender.Systinet_AUDIT.layout.ConversionPattern="%d",%X{audit.category}:%X{audit.eventId},
   %X{audit.result},%X{audit.ctxId},"%X{audit.actor}","%X{audit.resource}",%X{audit.detail}%n

# configure audit logging
log4j.category.com.hp.systinet.audit.event=DEBUG,Systinet_AUDIT
log4j.additivity.com.hp.systinet.audit.event=true

# limit categories that are too verbose
log4j.category.org.apache.xml.security=ERROR,file,stdout
```

```
log4j.additivity.org.apache.xml.security=true
log4j.category.org.hibernate=ERROR,stdout,file
log4j.additivity.org.hibernate=true
```

This configuration instructs log4j to do the following:

1 Print information,warning, and error messages to the console, and to a file named `log4j.log`, for all logging categories that are not explicitly declared.

SOA Systinet also uses the logging categories which start with one of the following:

- com.hp.systinet
- org.hp.systinet
- com.systinet
- org.systinet

2 Print the audit log to a file named `hpsoa_audit.log`

The format of the log is specified in the *`log4j.appender.Systinet_AUDIT.layout.ConversionPattern`* property in Example 2 on page 144. Each audit event is a single line that starts with date and time (formatted according to ISO8601), followed by comma separated attributes of the event.

3 A deployed SOA Systinet creates the log files in the following locations:

- **For JBoss:**

  `JBOSS_HOME/server/PROFILE_NAME/log`

- **For OAS:**

  `OAS_HOME/j2ee/OC4J_INSTANCE_NAME`

- **For WebLogic:**

  `DOMAIN_HOME`

`DOMAIN_HOME` is the location set in Creating Domains for SOA Systinet on page 75.

- **For WebSphere:**

  `PROFILE_HOME`

`PROFILE_HOME` is the location set in Creating a Profile on page 90.

4 The logging category, *`com.hp.systinet.audit.event`*, is used to log audit events. This logging category also has subcategories according to the audit event category. For example, the logging category name for audit events in the *licensing* category is *`com.hp.systinet.audit.event.licensing`*.

You can change the output or strip down the audit log for any particular audit category.

5 The other declared logging categories (hibernate, apache xml security) are stripped to only log error messages. These categories are too verbose for printing if information messages are also logged (the default for all categories).

Enabling Full Text Search

The SOA Systinet full text search is an optional feature based on relational database extensions.

To enable FTS:

1 Prepare FTS on the database server:

a Create an index for column "m_extension" from "ry_resource" table.

b Create an index for column "data" from "ry_resource" table.

c Schedule update of these indexes.

2 Activate FTS in the SOA Systinet UI, as described in the *SOA Systinet Configuration Options* section in the *HP SOA Systinet Administrator Guide* .

The following sections provide details on enabling Full Text Search:

- Enabling Full Text Search on DB2 on page 147
- Enabling Full Text Search on MSSQL on page 148
- Enabling Full Text Search on Oracle on page 150

Make sure that your database server meets the system requirements described in the Preparing Databases on page 35.

Enabling Full Text Search on DB2

To enable full text search you must create indexes and schedule their update in DB2. Use the DB2 Net Search Extender. Connect to the database using the same credentials used during installation.

Follow Example 3 on page 147.

Example 3: Create Indexes for FTS and Schedule Synchronization in DB2

```
db2text START

#use sa user in this case
db2text ENABLE DATABASE FOR TEXT CONNECT TO <database> USER sa USING <password>

db2text CREATE INDEX idx_ry_resource_meta FOR TEXT ON ry_resource(m_extensions)
  CONNECT TO <database> USER <user> USING <password>

db2text CREATE INDEX idx_ry_resource_data FOR TEXT ON ry_resource(data)
  CONNECT TO <database> USER <user> USING <password>

#schedule a regular index update each day at midnight
db2text ALTER INDEX idx_ry_resource_meta FOR TEXT UPDATE FREQUENCY D(*) H(0) M(0)
  CONNECT TO <database> USER <user> USING <password>

db2text ALTER INDEX idx_ry_resource_data FOR TEXT UPDATE FREQUENCY D(*) H(0) M(0)
  CONNECT TO <database> USER <user> USING <password>
```

Commands to update the index manually can be found in Example 4 on page 148.

Example 4: Synchronizing Indexes in DB2 Manually

```
db2text UPDATE INDEX idx_ry_resource_meta FOR TEXT
  CONNECT TO <database> USER <user> USING <password>

db2text UPDATE INDEX idx_ry_resource_data FOR TEXT
  CONNECT TO <database> USER <user> USING <password>
```

For more scheduling details, see the *DB2 Net Search Extender* documentation.

Enabling Full Text Search on MSSQL

To enable full text search you must enable the service and create a full text catalog and indexes. Use MSSQL Server Management Studio or the sqlcmd command line tool.

Connect to the database using the same parameters used during SOA Systinet installation.

To enable full text search on MSSQL:

1 Make sure that the SQL Server Fulltext Search service is running, and that the database is full-text enabled.

By default, new databases are full-text enabled unless you create them with MSSQL Server Management Studio.

In this case, select the database in the Object Explorer window, and select **Properties→Files**, and then select **Use full-text indexing**.

2 To create a full-text catalog, execute the following command:

```
sqlcmd -U <user> -P <password> -d <database>
CREATE FULLTEXT CATALOG ry_resource_ftsc
go
```

You must have CREATE FULLTEXT CATALOG permission.

It is possible to reuse an existing catalog, but HP Software recommend creating a new one for independent management purposes.

For more details, see http://msdn2.microsoft.com/en-us/library/ms189520.aspx.

3 Do one of the following:

- To create a full-text index that is synchronized immediately after any data changes, execute the following command:

```
sqlcmd -U <user> -P <password> -d <database>
CREATE FULLTEXT INDEX ON ry_resource(
    m_extensions TYPE COLUMN m_extensions_fe LANGUAGE 0x0,
    data TYPE COLUMN data_fe LANGUAGE 0x0)
KEY INDEX pk_resource_11 ON ry_resource_ftsc WITH CHANGE_TRACKING AUTO
go
```

- To create a full-text index that is synchronized manually, execute the following command:

```
sqlcmd -U <user> -P <password> -d <database>
CREATE FULLTEXT INDEX ON ry_resource(
    m_extensions TYPE COLUMN m_extensions_fe LANGUAGE 0x0,
    data TYPE COLUMN data_fe LANGUAGE 0x0)
KEY INDEX pk_resource_11 ON ry_resource_ftsc WITH CHANGE_TRACKING OFF, NO POPULATION
go
```

For more details, see http://msdn2.microsoft.com/en-us/library/ms187317.aspx.

To synchronize the index manually, execute the following command:

```
sqlcmd -U <user> -P <password> -d <database>
ALTER FULLTEXT INDEX ON ry_resource START FULL POPULATION
go
```

The statement executes asynchronously, so the population may take some time.

To verify the population status, execute the command:

```
SELECT FULLTEXTCATALOGPROPERTY('ry_resource_ftsc','PopulateStatus')
    go
```

Index population is complete when the population status is 0.

For more details, see http://msdn.microsoft.com/en-us/library/ms188359.aspx.

Enabling Full Text Search on Oracle

To enable full text search, you must create indexes and schedule their update. Use the Oracle **sqlplus** console. Connect to the database using the same credentials used during installation.

The procedure in commands is shown in Example 5 on page 150. It also shows how to synchronize indexes every midnight.

The database user does not have permission to create FTS indexes by default and must be given that permission.

Example 5: Preparing Oracle For Full Text Search using the Scheduling Mechanism

```
sqlplus system/password@connect_identifier
-- add permission to create indexes
GRANT EXECUTE ON "CTXSYS"."CTX_DDL" TO user;
-- add "create job" permission to <user>
GRANT CREATE JOB TO user;
exit;

sqlplus user/password@connect_identifier
CREATE INDEX idx_ry_resource_meta ON ry_resource(m_extensions)
    INDEXTYPE IS CTXSYS.CONTEXT PARAMETERS
        ('FILTER CTXSYS.NULL_FILTER SECTION
          GROUP CTXSYS.NULL_SECTION_GROUP
          SYNC (EVERY "TRUNC(SYSDATE)+1")');

CREATE INDEX idx_ry_resource_data ON ry_resource(data)
    INDEXTYPE IS CTXSYS.CONTEXT PARAMETERS
        ('FILTER CTXSYS.NULL_FILTER SECTION
          GROUP CTXSYS.NULL_SECTION_GROUP
          SYNC (EVERY "TRUNC(SYSDATE)+1")');
```

To enable full text search of pdf, doc, and other document types, use `AUTO_FILTER` in the definition of the `idx_ry_resource_data` index"

```
CREATE INDEX idx_ry_resource_data ON ry_resource(data)
    INDEXTYPE IS CTXSYS.CONTEXT PARAMETERS
    ('FILTER CTXSYS.AUTO_FILTER');
```

Not all document types can be indexed correctly.

For details, see http://download.oracle.com/docs/cd/B19306_01/text.102/b14218/afilsupt.-htm#i634493.

When you create the index, remove words that could frequently appear in full-text searches from the Oracle stoplist. By default, the Oracle index stoplist includes words such as "to." Full-text searches including these words return a false empty list. Alternatively, the database administrator should provide SOA Systinet users with the stoplist, and a warning not to use these terms in full-text searches.

Commands to set up the indexing stoplist on Oracle and to replace the Oracle stoplist are shown in Example 6 on page 151.

Example 6: Creating Oracle Indexing Stoplist

```
call CTX_DDL.CREATE_STOPLIST('MyStoplist');
call CTX_DDL.ADD_STOPWORD('MyStoplist', 'a');

... Add words that should not be indexed. Do not include "to" or other words that would frequently
        occur in full-text searches.

-- Include the DROP INDEX commands only if an index already exists.
DROP INDEX idx_ry_resource_meta;
DROP INDEX idx_ry_resource_data;
CREATE INDEX idx_ry_resource_meta on ry_resource(m_extensions) indextype is ctxsys.context parameters
        ('filter ctxsys.null_filter section group CTXSYS.NULL_SECTION_GROUP STOPLIST MyStoplist SYNC
(EVERY "TRUNC(SYSDATE)+1")') ;
CREATE INDEX idx_ry_resource_data on ry_resource(data) indextype is ctxsys.context parameters
        ('filter ctxsys.null_filter section group CTXSYS.NULL_SECTION_GROUP STOPLIST MyStoplist SYNC
(EVERY "TRUNC(SYSDATE)+1")');
```

For more information about creating indexes, see the Oracle documentation at http://download-uk.oracle.-com/docs/cd/B19306_01/text.102/b14218/toc.htm

Do not implement index synchronization ON COMMIT. It can cause Oracle thread termination, returning the error message `ORA-error stack (07445[ACCESS_VIOLATION]) logged in filename.log`. (Tested on Oracle 10gR2 - 10.2.0.1). You can use any other synchronization technique.

Executing index synchronization manually is shown in Example 7 on page 152.

Example 7: Synchronizing Indexes in Oracle Manually

```
sqlplus user/password@connect_identifier
CALL CTX_DDL.SYNC_INDEX('idx_ry_resource_meta', '2M');
CALL CTX_DDL.SYNC_INDEX('idx_ry_resource_data', '2M');
```

SSL Certificates

SOA Systinet can use HTTPS for communicating with other applications, such as UDDI registries, *HP Business Availability Center*, and *HP SOA Manager*, or to publish resources, such as WSDL and XML schemas, accessible via HTTPS.

HTTPS runs on top of SSL and requires additional setup steps. SSL requirements are driven by the target server, which can operate in one of the following modes:

1 **One-way SSL**

Mostly used for WWW, where the server authenticates to the client by providing its certificate.

The following conditions must be met:

- The server certificate must be valid and trusted.
- The resource HTTPS URL must contain a host name contained in the target server SSL certificate.

2 **Required two-way SSL**

Mutual authentication is required and both server and client authenticate each other using certificates.

The following conditions must be met:

- The requirements for one-way SSL.
- The client must provide its certificate, and the server must trust the client certificate.

3 **Wanted two-way SSL**

Mutual authentication is wanted but not mandatory and the connection can be established as one-way SSL.

The requirements are the same as for one-way SSL.

Using SSL is described in the following sections:

Identifying Server SSL Requirements

SOA Systinet deployed to an application server acts as an HTTPS client.

The SSL requirements of the target server can be determined using the ssltool provided with SOA Systinet.

For example, with the command:

ssltool serverInfo --url https://localhost:8443 --certFile localhost.crt

The output resembles the following:

```
localhost:2968
Host: localhost/127.0.0.1
Port: 2968
```

```
Client authentication: WANTED
Accepted issuers of client certificates:
 OU=Some CA, CN=CA
Server Certificate:
 Subject: CN=localhost
 Issuer: OU=Some CA, CN=CA
 Issued for hostname(s): localhost
 Algorithm: RSA
 Valid from: 12/7/07 4:22 PM
 Valid to: 12/4/17 4:22 PM
Received Server Certificate Chain:
 CN=localhost
 OU=Some CA, CN=CA
PEM Encoded Received Server Certificate Chain:
-----BEGIN CERTIFICATE-----
  ...
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
  ...
-----END CERTIFICATE-----
```

The output contains the following elements:

- The SSL server mode is output after `Client Authentication`, where:
  - `NOT_REQUESTED` means one-way SSL.
  - `WANTED` means non-mandatory two-way SSL.
  - `REQUIRED` means required two-way SSL.
- The accepted issuers for two-way SSL. These display the subject names of accepted certificate authorities that can issue client certificates.
- The server certificate details saved to a file, `localhost.crt`.
- Host names that must be used in URLs are listed after `Issued for hostname(s):`.
- PEM encoded certificates that must be trusted.

For more details about the SSL Tool, see "SSL Tool" in the *HP SOA Systinet Administration Guide* .

SSL Server Certificate Trust

By default, any SSL server certificate is trusted.

For non-default options, see SSL Customization on page 155.

Trusting HP SOA Registry Foundation Certificates

If you are using SOA Systinet integrated with HP SOA Registry Foundation and intend to import or export artifacts and taxonomies, then SOA Systinet must trust the HP SOA Registry Foundation certificate.

For details, see SSL Server Certificate Trust on page 155.

Importing Client Certificates for Two-Way SSL

A client certificate (a private key with an associated certificate) issued by a certificate authority trusted by the server must be imported to the keystore. By default, this keystore is the one used by JDK. Usually it is empty or unspecified. The location of the JDK SSL keystore is determined using a procedure explained at http://java.sun.com/j2se/1.5.0/docs/guide/security/jsse/JSSERefGuide.html#X509KeyManager. JDK keystores are managed by the keytool.

For details about keytool, see http://java.sun.com/j2se/1.5.0/docs/tooldocs/index.html#security.

For non-default options, see SSL Customization on page 155.

SSL Customization

SOA Systinet customizes SSL trust, key management, and hostname verification. There are four available SSL customizations available after installation. The default customization is named `skipped` and this is the customization used in the previous procedures in this section.

The list of available customizations is available with the command:

ssltool customize --info

The output should resemble the following:

```
Effective customization: composite
Available customizations:
--> default
  Java/JSSE default key/trust stores, default hostname verifier.
--> skipped
  Server certificates are always trusted and pass hostname verification, default
  key manager is used.
--> database
  Database key/trust stores, default hostname verifier.
--> composite
  Composition of database and default key/trust stores, default hostname verifier.
```

- **default**

  This means no customization and the application server trust and keystore is used.

  Hostname verification is required.

- **skipped**

  This means that server certificates are ignored as no trust is required.

  Hostname verification is ignored.

  Client certificates are taken from a file described here at http://java.sun.com/j2se/1.5.-0/docs/guide/security/jsse/JSSERefGuide.html#X509KeyManager.

  The `skipped` customization is a default, it is chosen during installation by doing one of the following:

  - In the GUI Installation wizard, Endpoint Properties page, deselect **Verify Certificates**.

    For details, see Using the GUI Installer on page 19 Step 16.

  - In the `configuration.properties` file, set the following property:

    ```
    platform.certVerification=skipped
    ```

    For details, see Configuring the Endpoint on page 119.

- **database**

This means that only the SOA Systinet database is used for the trust and keystore.

Hostname verification is required.

- **composite**

  This means that a composite of the JDK truststore and SOA Systinet database truststore is used.

  A certificate is trusted if it is in either the JDK or the SOA Systinet database truststore.

  Client certificates are taken from both the JDK and SOA Systinet database keystores.

  Hostname verification is required.

  The `composite` customization can be chosen during installation by doing one of the following:

  - In the GUI Installation wizard, Endpoint Properties page, select **Verify Certificates**.

    For details, see Using the GUI Installer on page 19 Step 16.

  - In the `configuration.properties` file, set the following property:

    ```
    platform.certVerification=composite
    ```

    For details, see Configuring the Endpoint on page 119.

You can manage the database keystore and truststore using the ssltool provided with SOA Systinet. These stores are specific to SOA Systinet. Unlike the JDK keystore, changing the database keystore does not require a restart of the application server, because changes take effect in approximately thirty seconds.

To import a server certificate to the database truststore:

- Execute the following command:

  SOA_HOME/bin/ssltool keystoreEI -i truststore --certfile *CERTIFICATE_FILE* --add

To import a client certificate to the database keystore:

- Execute the following command:

SOA_HOME/bin/ssltool keystoreEI -i keystore --keystore *CLIENT_CERT* --storepass *PASSWORD* --add

CLIENT_CERT can be either a Java keystore with a private key or a p12 file (PKCS 12). *PASSWORD* is the password used to protect the file.

For more details about the SSL Tool, see "SSL Tool" in the *HP SOA Systinet Administration Guide* .

To change the SSL customization:

- Do one of the following:
  - Change the setting in the SOA Systinet Configuration page.

    For details, see "Configuration Options" in the *HP SOA Systinet Administration Guide* .
  - Execute the following command:

    SOA_HOME/bin/ssltool customize --change *CUSTOMIZATION_NAME*

SSL Troubleshooting

You can test the SSL setup for a particular HTTPS URL using the self-tester.

The test page is available at the following URL:

http://localhost:port/context/self-test/self-http-test

Input the URL to check and click **Test Connection**.

The output resembles the following:

For more details about the self-tester, see SOA Systinet Self-Tester on page 160.

Configuring LDAP over SSL/TLS

You can configure LDAP over *SSL* (or*TLS*) with a directory server of your choice. HP Software recommend that you first install SOA Systinet with a connection to LDAP that does not use SSL. You can then verify the configuration by logging in as a user defined in this directory before configuring use of SSL.

The configuration procedure assumes that you have already installed SOA Systinet with an LDAP account provider.

SOA Systinet must not be running.

LDAP over SSL Without Client Authentication

In this case only LDAP server authentication is required. This is usually the case.

To change the LDAP configuration, run the Setup Tool and change **Naming Provider URL** to use the `ldaps` protocol and the port on which the directory server accepts SSL/TLS connections. An example of such a URL is, `ldaps://ldap.test.com:636`.

Make sure that the hostname specified in the `java.naming.provider.url` property matches the name that is in the directory server certificate's subject common name (CN part of certificate's Subject). Otherwise you get an exception during startup of SOA Systinet. It informs you of a hostname verification error. The stacktrace contains the hostname that you must use.

LDAP over SSL With Mutual Authentication

SOA Systinet does not support LDAP over SSL with mutual authentication.

Ensuring Trust with the LDAP Server

The client that connects to the SSL/TLS server must trust the server certificate in order to establish communication with that server. The configuration of LDAP described in this section inherits the default rule for establishing trust from JSSE (the Java implementation of SSL/TLS). This is based on trust stores.

For details, see SSL Server Certificate Trust on page 155.

SOA Systinet Self-Tester

SOA Systinet includes a self-tester that checks various aspects of deployment. The self-tester runs automatically every time SOA Systinet is started.

If there are deployment problems and the bundled self-tester does not function, you can deploy the self-tester as a standalone application.

For details see, Stand-Alone Self-Test Deployment on page 161.

As a security measure, you should disable the self-test output after successful deployment.

For details, see Turning Off Self-Test Output on page 162.

The self-tester performs the following checks:

Table 13. Self-Tests

Self-Test	Description
Product configuration checks	Checks product configuration, versions, and libraries.
Product runtime checks	Checks logging configuration, and outputs product base URLs.
Application server checks	Checks application server and JVM settings.
JNDI checks	Checks required JNDI resources.
Datasource checks	Checks the data source connection.
JMS checks	Checks the sending of JMS messages to required JMS destinations.

You can view the self-test results in the server output console or with your browser.

In the default configuration, the server console output includes only information about the groups of checks that are run and any errors that occur.

The web output is more informative and readable, showing all the checks run and the results.

Access the self-test output at the following URL:

`http://hostname:port/context/self-test`

If errors occur, the self-tester provides details about the errors and suggests how to solve the underlying problems.

Self-test also enables you to test HTTP/HTTPS connections to simulate access to external resources in the same way as a deployed SOA Systinet. Access this feature at the following URL:

`http://hostname:port/context/self-test/self-http-test`

Stand-Alone Self-Test Deployment

In cases where the self-tester bundled with SOA Systinet cannot be accessed, for example, when the server is badly configured and SOA Systinet cannot be correctly deployed, you can deploy the self-tester as a standalone application.

The package is prepared for deployment in, `SOA_HOME/deploy/self-test-standalone.war`.

Deploy the WAR file using the functionality of your application server or copy the WAR file to your JBoss deploy directory.

If the SOA Systinet EAR is not deployed, all the product runtime checks fail.

To execute the stand-alone self-tester and access its output, access the following URL:

`http://hostname:port/self-test`

The self-test context can be changed in most application servers.

Turning Off Self-Test Output

The self-test output is accessible to everyone. For security reasons, switch off the self-test output after a completed deployment of SOA Systinet passes the self-test.

To switch off the self-test output:

1. Sign in to SOA Systinet as the administrator.
2. In the Tools tab Administration menu, click **Configuration**.

   The Configure page opens.
3. In the Configure page, deselect **Self Test Access**.
4. Click **Save**.

To disable the standalone self-tester, undeploy the `self-test-standalone.war` package from your server.

To verify that the self-test has been disabled, check the self-test output URL.