Subscription Repository

Installation and Administration Reference

v.7.0

Legal Notices

Table of Contents

In This Guide

This guide explains how to manage and configure the performance of the Subscription Repository (SR) system.

Audience

The audience for this guide is the Solutions Integrator (SI). The SI has a combination of some or all of the following capabilities:

- Understands and has a solid working knowledge of:
  - UNIX® commands
  - Windows® system administration
- Understands networking concepts and language
- Is able to program in Java™ and XML
- Understands security issues
- Understands the customer's problem domain

References

Manual Organization

This guide contains the following chapters:

- Chapter 1, "Introduction", provides a brief explanation about the purpose, the scope and the definitions involved in this document.
- Chapter 2, "System Architecture", provides a description of its architecture.
- Chapter 3, "Basic Operations", gives a general description of the basic system management.
- Chapter 4, "Installation", describes the installation process.
- Chapter 5, "Configuration", describes the configuration process and performance options.
- Chapter 6, "Security and authorization", describes how to handle with them.
- Chapter 7, "Troubleshooting", list of possible troubles and how to solve them.
- Chapter 8, "General Performance Recommendations", performance recommendations to be followed.

Conventions

The following typographical conventions are used in this guide.

Font	What the Font Represents
Italic	Book or manual titles, and man page names
	Provides emphasis
	Specifies a variable that you must supply when entering a command
	Parameters to a method
Bold	New terms
`Computer`	Text and items on the computer screen
	Command names
	Method names
	File and directory names
	Process names
	Window/dialog box names
	XML tag references
`Computer Bold`	Text that you must type
Keycap	Keyboard keys
[Button]	Buttons on the user interface
Menu Items	A menu name followed by a colon (:) means that you select the menu, then the item. When the item is followed by an arrow (->), a cascading menu follows

Install Location Descriptors

The following names are used throughout this guide to define install locations.

Descriptor	What the Descriptor Represents
`$JBOSS_HOME`	The install location for JBoss. The location is depending on the path defined in installation process (i.e. <drive>:\jboss).
`$SOLR_HOME`	The install location for Solr. The location is depending on the path defined in installation process (i.e. <drive>:\solr).

1. Introduction

1.1. Purpose

This document is a manual for Subscription Repository Administrators. It explains how to manage and configure the Subscription Repository system.

1.2. Document Scope

This document is oriented for Subscription Repository Administrators.

1.3. Definitions

1.3.1. Acronyms

- SR: Subscription Repository
- DBMS: Database Management System
- SID: Shared Information Model
- CRUD: Creation Read Update Delete
- OM: Order Management
- WS: Web Service
- WSDL: Web Service Definition Language
- CFS: Customer Facing Service
- RFS: Resource Facing Service

2. System Architecture

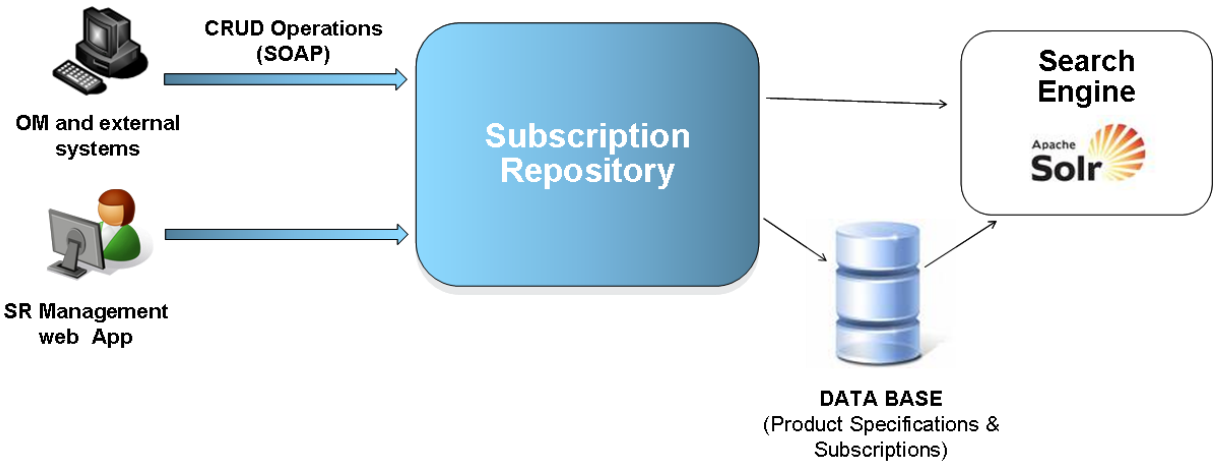
This chapter gives a general description of the architecture of the SR Application.

The SR application is a repository for subscribed products based on SID. It's written in java and runs as a standalone search server within a servlet container. It uses a Database to store the product information and Solr (enterprise search platform from Apache) for providing advanced searching mechanisms.

Based on SID, it stores two kinds of information:

- **Product Specifications**. Describe the structure of one product, including the services and resources that this product contains, and the different characteristics for each of them.
- **Product Subscriptions**. Main elements of the system. They contain specific information about a concrete product acquired by a customer. They follow the structure defined in the specification.

This diagram shows an overview of the SR Application.



The main parts of SR Application are:

- **CRUD operations**. Basic read, create, update and remove operations (for product specifications and subscriptions). Exposed as web services. Product information is structured following the SID principles.
- **Advanced Search**. Execute advanced searches against the system. The information in the database is indexed in a Search Engine for this purpose.
- **External Processes**. Possibility to define java 'plugins' that interact with the system to export information to other systems and migrate the existing products.
- **Management Application**. A GUI is provided to manage the SR system.

3. Basic Operations

3.1. Start Subscription Repository System

There are two script files available that start the system in Windows OS or in a UNIX-based OS. Use the command `'$JBOSS_HOME/run.bat -b 0.0.0.0'` to start the SR system in Windows or `'$JBOSS_HOME/run.sh -b 0.0.0.0'` for UNIX-based Operating Systems.

In addition you can configure the SR as a system service. For additional information consult JBoss official reference [http://docs.jboss.org/jbossas/docs/Installation_And_Getting_Started_Guide/5/html/The_JBoss_Server__A_Quick_Tour.html#Starting_and_Stopping_the_Server].

3.2. Stop Subscription Repository System

If the SR process is running in an active window, it can be stopped using **Ctrl + C**.

However, there are two script files available that stop the system in Windows OS or in a UNIX-based OS. Use the script `$JBOSS_HOME/shutdown.bat` to stop the SR system in Windows or `$JBOSS_HOME/shutdown.sh` for UNIX-based Operating Systems.

For additional information consult JBoss official reference [http://docs.jboss.org/jbossas/docs/Installation_And_Getting_Started_Guide/5/html/The_JBoss_Server__A_Quick_Tour.html#Starting_and_Stopping_the_Server].

3.3. Check Subscription Repository System

In order to check if the system is running properly you can try to access to two locations.

`http://<ip>:8080/solr` - This URL allows checking if the Solr search engine is working properly. The message 'Welcome to Solr!' appears if Solr is up and running.

`http://<ip>:8080/subscriptionrepository` - This URL allows checking if the SR system is working properly. The message 'Welcome to Subscription Repository!' appears if SR is up and running.

3.4. Check traces

Log files are available in the path `$JBOSS_HOME/server/default/log/server.log`.

4. Installation

The installation process only unzips a prepared distribution of JBoss and Solr. But, as it is adviced in a script message, the configuration process is absolutely necessary for the SR system to work successfully. For this reason the script will ask you if you want to execute the configuration process after the installation process.

4.1. Requirements

- Java distribution installed in local machine (version 1.6).
- Windows 2008 Server, Red Hat Enterprise Linux 6, HP-UX.
- Accessible DBMS (Oracle 11g or 12c , MySQL 5.1 or Postgres Plus A.S. 9.2) up and running.
- Database client (sqlplus.exe, mysql.exe or edbplus.bat) available in local machine and compatible with the selected DBMS.
- At least 1 GB of RAM memory available only for JVM (fix this value in JBoss run script).
- Space estimation for 10 millions and 30 millions of subscriptions:

	10 MILLIONS	30 MILLIONS
DATABASE*	30 GB	90 GB
SOLR INDEX**	13 GB + 13 GB	40 GB + 40 GB
TOTAL	***56 GB***	***170 GB***

NOTE: Read "*HP Subscription Repository – Performance.doc*" for more detailed information.
NOTE*: In case of database, size required is not very important. New indexation process uses a configurable batch size, so needed space for results is small (compared to total database size).
NOTE:** You must have a free space of at least the same size occupied by the Solr Index. This is necessary for the optimization task (optimization the Solr index), and is the reason for the duplication of Solr Index required size.

4.2. Recommendations

- Performance is optimized for 30 millions of subscriptions. However, the complexity of information could decrease this performance. In general, if the estimate volume is greater than 30 millions it is recommended to mount a cluster at least at Solr level.
- Set as much RAM as possible only for JVM (fix this value in JBoss run script).

4.3. Installation with Script Assistant

In order to properly install the SR system, follow these steps:

- Unzip the installer package.
- Execute the script `install.bat` in Windows OS or `install.sh` in UNIX-based operating systems. During the execution the script could request some information (see section 'Installation Parameters' in this chapter).

4.4. Installation with Script Assistant, without sql client (special case)

It´s installation special case when user has not the sql client (for example: sqlplus) in the same server where SR is going to be installed. To evict this problem, script could be executed in two machines in different ways:

1. Script installation for SR machine:
    a. Unzip the installer package.
    b. Execute the script `install.bat` in Windows OS or `install.sh` in operative systems like UNIX.
    c. During the script execution, first step is Jboss and Solr Jboss server installation. After application configuration step will be executed. For this, user must be select "Y" for question: "[?] Do you want to configure OM Subscription Repository right now? (Y/N):". When "SQLPlus binary", "MySQL monitor binary" or "EDBPlus binary" is required, user should introduce one existing file path (for instance: C:\sr-installer\lib\install.js). In next question "[?] Do you want to (re)create the database structure now? (Y/N): ") installation process must be stopped (with Ctrl+C).

    Script output could be:

```
C:\sr-installer>install.bat
Welcome to the OM Subscription Repository Installer

[i] No configuration file found.

[i] You must answer the following questions in order to install the product.
[i] Pressing the ENTER key accepts the default value (between []) if any.

[?] JBoss home destination directory []: C:\SR\jboss
[?] Solr home destination directory []: C:\SR\solr

[i] Installing JBoss data...
[i] JBoss data successfully installed.
[i] Installing Solr data...
[i] The chosen destination directory is not empty.
[?] Do you want to clean this directory before copying new files? (Y/N): Y
[i] Cleaning destination directory...
[i] Done. Installing...
[i] Solr data successfully installed.
[i] Installing InstantOn license...
log4j:WARN No appenders could be found for logger (com.hp.autopassj.core.config.
AutopassJConfigData).
log4j:WARN Please initialize the log4j system properly.
[i] InstantOn license successfully installed.

[i] OM Subscription Repository has been installed.
[i] However, it needs to be configured before first run.
[?] Do you want to configure OM Subscription Repository right now? (Y/N): Y

[i] Next, the installed product will be configured for use.
[i] Please answer the following questions in order to configure the product.
[i] Pressing the ENTER key accepts the default value (between []) if any.

[?] Java home for JBoss to use (JRE/JDK 1.6+) [C:\java\jdk1.6.0_25\jre]:
[?] Type of DBMS to use (mysql/oracle/ppas) []: mysql
[?] Database user name []: sruser
[?] Database user password: ¡foe09.leod
[?] Database host name [localhost]: 192.168.1.25
[?] Database port [3306]: 3306
[?] Database schema []: sr
[?] Do you want to enable debugging in the JBoss instance (Y/N) []: N
```

```
[?] Do you want to enable secure connections in the JBoss instance (Y/N) []: N
[i] Securizing server...
[i] Security disabled (https).

[?] Complete path to the MySQL monitor binary []: C:\sr-installer\lib\install.js

[i] Processing configuration files...
[i] Processing 'C:\sr\jboss\bin\run.conf.bat'...
[i] Processing 'C:\sr\jboss\bin\run.conf'...
[i] Processing 'C:\sr\jboss\server\default\conf\bootstrap\aop.xml'...
[i] Processing 'C:\sr\jboss\server\default\conf\login-config.xml'...
[i] Processing 'C:\sr\jboss\server\default\deploy\subscriptionrepository-ds.xml
'...
[i] Processing 'C:\sr\jboss\server\default\deploy\subscriptionrepository.war\WE
B-INF\web.xml'...

[i] OM Subscription Repository has been configured.
[i] For it to function correctly, the database structure must be (re)created.
[i] If you choose to do so now, all previous schema information will be lost.
[?] Do you want to (re)create the database structure now? (Y/N): Terminate batch job (Y/N)?Y

[Ctrl+C]

C:\ sr-installer >
```

With previous step, SR has been installed correctly.

2. Script installation in database machine.
    a. Unzip the installer package.
    b. Execute the script `install.bat` in Windows OS or `install.sh` in operative systems like UNIX. Execute fully installation. During installation, user must introduce correct configuration values. It´s very important that executes "all database steps".
    c. When installation process ends, user must delete jboss and solr directories of database machine:

```
…
[?] JBoss home destination directory []: C:\SR\jboss
[?] Solr home destination directory []: C:\SR\solr
…
```

With first installation process, user has installed (deploy and do static configuration) SR. Second installation process has complete all database installation (create structure and introduce dynamic configuration). Remember that all configuration data introduced during second installation:

```
[i] In this section user has to introduce basic configuration parameters of SR.

[?] Default max size of result list [100]:
[?] Do you want to enable subscription history system (save copies of each subscription modified or deleted.) (Y/N) [N]: N
[?] Do you want to enter Quartz indexation process trigger expression (to use in dexation offline) (Y/N) []: N
[?] Do you want to enter Quartz specification indexation process trigger expression (to use indexation offline) (Y/N) []: N
[?] Do you want to enter Quartz index optimization process trigger expression (Y/N) []: N
[i] Next question is about indexation level. Available levels are:
[i] LEVEL1 : Only characteristicName_type pattern is used.
[i] LEVEL2 : The componentName_characteristicName_type pattern and LEVEL1Æs pattern are used.
[i] LEVEL3 : The componentName_componentVersion_characteristicName_type pattern and LEVEL2Æs patterns are used.
[?] Indexation level [LEVEL1]:
```

It applies for the first installation (because these data are saved into database).

Only it´s required manual operation when SR server is running in cluster. In this case, user must read section 5.3.4 , because cluster configuration in "web.xml" file is last step of installation and is not done in first described installation.

Remember that introduced database configuration data refers always to same database service, but could not be the same in both installation processes. It´s a good idea to use public IP address (not localhost). Besides, this installation case is valid only for simple SR installation, where SR and Solr are installed in same server.

4.5. Installation Solr Cluster

In order to properly install the SR system with a Solr cluster, follow these steps:

- Unzip the installer package for each node.
- Execute on the master node the script `install.bat` in Windows OS or `install.sh` in UNIX-based operating systems. During its execution the script it could request some information (see section 'Installation Parameters' in this chapter).
- Execute on the rest of nodes the script `installSolrNode.bat` in Windows OS or `installSolrNode.sh` in UNIX-based operating systems. During the script execution it could request some information (see section 'Installation Parameters' in this chapter).
- Configure the parameter 'Solr URLs' (see subsection 'Application Configuration File' in next chapter) in master node with the list of URLs, using as first URL the corresponding to the master node.

4.5.1. Add Node

If only want to add a new Solr node to a previous installation. This operation generate that system down in a severe inconsistence state (see subsection 'Severe Inconsistences' in chapter 'Troubleshooting'). In pre-production environment the system will have an additional node, for this reason, an additional Solr node will be installed in production environment previous to the overwriting of data folders and the system configuration will be fixed with the additional URL in parameter 'Solr URLs' (see subsection 'Application Configuration File' in next chapter).

4.5.2. Remove Node

If only want to remove a new Solr node from a previous installation. This operation generate that system down in a severe inconsistence state (see subsection 'Severe Inconsistences' in chapter 'Troubleshooting'). In pre-production environment the system will have less nodes than current production environment, for this reason, a Solr node will be removed in production environment previous to the overwriting of data folders and the system configuration will be fixed with the removed URL in parameter 'Solr URLs' (see subsection 'Application Configuration File' in next chapter).

4.6. Installation Parameters

This information is required in order to obtain a successfully installation.

Parameter key	**Description**	**Default value**
jboss.home	jboss home	
solr.home	solr home	
java.home	java home for jboss	JAVA_HOME defined in

		system properties
ds.type	database type	
ds.oracle.version	Oracle database version (only required for ds.type=oracle)	11/12
ds.user	database user name	
ds.password	database user password	
ds.host	database hostname	localhost
ds.port	database port	1521 (oracle) / 3306 (mysql) / 5444 (postgress)
ds.sid	database SID (only for oracle)	
ds.schema	database Schema (only for mysql and ppas)	
jboss.debug	enable debug traces in jboss log	
dba.user	database administrator name	
dba.password	database administrator password	
path.to.sqlplus	sqlplus binary path	try to detect the sqlplus binary path by default
path.to.mysql	mysql binary path	try to detect the mysql binary path by default
path.to.edbplus	edbplus binary path	try to detect the edbplus binary path by default
Jboss.secure	Specified if SR conf for secure connections	
cluster.mode.active	SR has been configured for cluster	
config.max_results	Default response num. results	100
config.validation_type	Subscription validation type	STRONG
config.indexation_level	Characteristics indexation level	LEVEL1
config.indexation_mode	Characteristics indexation mode	INCLUSIVE
config. solr_indexing_scheduled.active	User has configured quartz expression	
config. solr_optimization_scheduled	Quartz trigger expression for process.	
config. solr_optimization_scheduled.active	User has configured quartz expression	
config. solr_indexing_scheduled	Quartz trigger expression for process.	
config. solr_specification_indexing_scheduled.active	User has configured quartz expression	
config. solr_specification_indexing_scheduled	Quartz trigger expression for process.	
history_system_active.active	Enable history system.	false

The installation script allows save all these parameters (except passwords information) in the `install.conf` file in order to use these values as default (overwriting default values in table) in the next installation.

4.7. Unattended Installation

SR can be installed automatically getting the parameters from the file (`install.conf`).

This is a configuration file example:

```
#Automatically saved from the OM Subscription Repository installation script.
#Fri Jul 12 12:42:44 CEST 2013
ds.host=localhost
cluster.mode.active=N
config.solr_optimization_scheduled=
config.solr_specifications_indexing_scheduled=
config.solr_indexing_scheduled.active=N
jboss.debug=N
path.to.edbplus=C\:\\Program Files\\PostgresPlus\\9.2AS\\edbplus\\edbplus.bat
jboss.secure=N
config.solr_optimization_scheduled.active=N
dba.user=enterprisedb
dba.password=PPASadmin1234
java.home=C\:\\java\\jdk1.6.0_24\\jre
config.validation_type=STRONG
config.indexation_level=LEVEL1
config.solr_indexing_scheduled=
config.history_system_active.active=N
config.max_results=10
ds.user=sruser_local
ds.password=sruser_local
config.indexation_mode=INCLUSIVE
ds.schema=sr_local
ds.sid=sr_local
ds.type=ppas
config.solr_specifications_indexing_scheduled.active=N
jboss.home=c\:\\SR_ppas_local\\jboss
solr.home=c\:\\SR_ppas_local\\solr
ds.port=5444
```

In order to install the SR include the parameter “unattended” in the command line.

- *install.bat unattended* (windows)
- *install.sh unattended* (unix)

5. Configuration

5.1. Reconfiguration using Script Assistant

Although the configuration process is part of the installation process, it can also be executed separately.

- Execute the script `configure.bat` in Windows OS or `configure.sh` in UNIX-based operating systems. During its execution the script could request some information (refer to section 'Installation Parameters' in previous chapter).

5.2. Configuration Files

5.2.1. Datasource Configuration Files

The datasource configuration is a task performed by the configuration process and it does not require any manual changes. There are two files implicated in datasource configuration:

- `$JBOSS_HOME/server/default/deploy/subscriptionrepository-ds.xml`
- `$JBOSS_HOME/server/default/conf/login-config.xml`

In the `login-config.xml` file are defined the credentials:

```
<!- Security domains for subscription repository -->
<application-policy name="SREncryptDBPassword1">
  <authentication>
    <login-module code="org.jboss.resource.security.SecureIdentityLoginModule" flag="required">
      <module-option name="userName">?username</module-option>
      <module-option name="password">?encrypted_password</module-option>
      <module-option name="managedConnectionFactoryName">
        jboss.jca:name=DefaultSR1,service=LocalTxCM
      </module-option>
    </login-module>
  </authentication>
</application-policy>
```

In the `subscriptionrepository-ds.xml` file is defined the datasource using the credentials defined in `login-config.xml` file.

```
<datasources>
  <local-tx-datasource>
    <jndi-name>DefaultSR1</jndi-name>
    <connection-url>?connection_string</connection-url>
    <driver-class>?driver_class</driver-class>
    <security-domain>SREncryptDBPassword1</security-domain>
    <exception-sorter-class-name>?exception_sorter_class</exception-sorter-class-name>
  </local-tx-datasource>
</datasources>
```

The main configurable variables are:

Variable	Description
?username	Username for database connection
?encrypted_password	Encrypted password for database connection (see below how encrypted password can be generated)

?connection_string	Connection string for the current instance or schema
?driver_class	Driver class for the current DBMS
?exception_sorter_class	Sorter class for the current DBMS

In order to obtain the encrypted password execute in Windows OS the command

```
java -cp "$JBOSS_HOME\client\jboss-logging-
spi.jar;$JBOSS_HOME\common\lib\jbosssx.jar"
org.jboss.resource.security.SecureIdentityLoginModule ?password
```

Or in UNIX-based Operating Systems the command

```
java -cp "$JBOSS_HOME/client/jboss-logging-
spi.jar:$JBOSS_HOME/common/lib/jbosssx.jar"
org.jboss.resource.security.SecureIdentityLoginModule ?password
```

For additional information consult JBoss official reference [http://docs.jboss.org/jbossas/docs/Administration_And_Configuration_Guide/5/html/Configuring_JDBC_DataSources.html].

5.2.2. Application Configuration File

The application configuration is a task partially done by the configuration process, but it requires some manual changes. There is one file implicated in application configuration:

- `$JBOSS_HOME/server/default/deploy/subscriptionrepository.war/WEB-INF/web.xml`

Besides, for cluster installations, some parts of the configuration will be defined at database level, specifically in the table *configuration*. This way the user can change max_number_results without rebooting the application. However other parameters cannot be modified or applied without performing a reboot. For instance, indexation level requires the user to reboot the app and reindex all data.

Only the following configuration parameters can be specified in configuration table: "max_number_results", "validation", "indexation_level", "indexation_mode", "external_processes_checking_interval", "unpluggued_indexation_block_size", "solr_indexing_scheduled", "solr_optimization_scheduled", "solr_specifications_indexing_scheduled", "solr_urls", "solr_urls_for_read", "solr_highlight_fields", "sr_urls", "all_data_specification_index", "history_system_active", "solr_indexing_files_enabled", "indexation_batch_size", "enable_related_subscriptions_indexation" can be specified only in the database.

(NOTE: application cannot check that database keys are unique, so you must check manually that only one key-value exists. In specific cases ("solr_urls" and "sr_urls"), it is possible to enter several rows with same key, one per server url).

In the `web.xml` file all the application configuration is defined:

```
<servlet>
  <servlet-name>subscription_repository</servlet-name>
  <servlet-class>com.hp.om.sr.servlet.SRServlet</servlet-class>
  <init-param>
    <param-name>inventory_type</param-name>
```

```
    <param-value>?inventory_type</param-value>
  </init-param>
  <init-param>
    <param-name>inventory_datasource</param-name>
    <param-value>?inventory_datasource</param-value>
  </init-param>
  <init-param>
    <param-name>search_engine_type</param-name>
    <param-value>?search_engine_type</param-value>
  </init-param>
  <init-param>
    <param-name>dynamic_loading_folders</param-name>
    <param-value>?search_engine_reindexing_mock_mode</param-value>
  </init-param>
  <init-param>
    <param-name>dynamic_loading_folders</param-name>
    <param-value>?dynamic_loading_folders</param-value>
  </init-param>
  <init-param>
    <param-name>solr_highlight_fields</param-name>
    <param-value>?solr_highlight_parameters </param-value>
  </init-param>
  <init-param>
    <param-name>security_authorize_status</param-name>
    <param-value>?security_authorization_status_parameters </param-value>
  </init-param>
  <init-param>
    <param-name>external_processes_checking_interval</param-name>
    <param-value>?external_processes_checking_interval</param-value>
  </init-param>
  <init-param>
    <param-name>unpluggued_indexation_block_size</param-name>
    <param-value>?unpluggued_indexation_block_size</param-value>
  </init-param>
  <init-param>
    <param-name> temp_folder </param-name>
    <param-value>?temp_folder_parameter </param-value>
  </init-param>
  <init-param>
    <param-name> all_data_specification_index </param-name>
    <param-value>?all_data_specification_index_parameter </param-value>
  </init-param>
  <load-on-startup>1</load-on-startup>
 </servlet>
```

Parameter Inventory Type

Defines the type of system which is supporting the inventory subsystem of the Subscription Repository. This parameter is set at configuration process according to values provided for datasource configuration.
The allowed values are *DB_ORACLE*, *DB_MYSQL* or *DB_PPAS*
The default value is *DB_ORACLE* if this parameter is not defined in `web.xml` file. This information has to be configured in web.xml file (not in database configuration table) because it is server-specific.

Parameter Inventory Datasource

Defines the datasource selected for the inventory subsystem to work properly. This parameter normally will not need any modification. If the value of this parameter is changed then many changes in other configuration files could be required.

The default value is *DefaultSR1*. This information has to be configured in web.xml file (not in database configuration table) because it is server-specific.

Parameter Max Number Results

Defines the limit for results in inventory queries and the limit for pagination in search engine. If an inventory query returns a number of results that is greater than this limit, then the system returns an error. In case of search engine it returns an error if the pagination size requested is greater than this limit.
The default value is *100* if this parameter is not defined in `web.xml` file or the value defined does not have number format.

Parameter External Processing Checking Interval

Defines the interval (in milliseconds) for checking the status of the process executed externally to the jvm.
The default value is 300000 ms (5 minutes) if this parameter is not defined in `web.xml` file or at database-level.

Parameter Unplugged Indexation Block Size

Defines the block size for queries executed by unplugged indexation process. This parameter only has effect when the system is working on DB_ORACLE inventory type.
The default value is 10000 if this parameter is not defined in `web.xml` file or at database-level.

Parameter Search Engine Type

Defines the type of search engine which is supporting the search engine subsystem of the Subscription Repository.
The allowed value is *SOLR*.
The default value is *SOLR* if this parameter is not defined in `web.xml` file.

Parameter Search Engine Reindexing Mock Mode

Defines if the system works in reindexing mock mode or not. While the system works in mock mode any change is done in database registers.
The default value is *false* if this parameter is not defined in `web.xml` file or the value defined is not *true*.

Parameter Solr Indexing Scheduled

Defines when the indexing process is executed using a cron expression, and in case this parameter has nott been defined or the provided value is not a valid cron expression, the indexing process works in on-line mode and it will be done at the same time for each operation in the inventory.
The allowed values are valid cron expressions (consult quartz reference [http://www.quartz-scheduler.org/docs/tutorials/crontrigger.html]).

Parameter Solr Optimization Scheduled

Defines when the optimization process is executed using a cron expression. This optimization process improves the system efficiency.
The allowed values are valid cron expressions (see quartz reference [http://www.quartz-scheduler.org/docs/tutorials/crontrigger.html]).
The default value is *"0 30 23 ? * SAT"* (every Saturday at 23:30) if this parameter is not defined in `web.xml` file or at database-level, or if the provided value is not a valid cron expression.

Parameter Solr URLs

Defines the URL or URLs (valid character separators: comma, semicolon and blank) where search engines Solr are available. The first Solr identified by its URL will work as master in queries.
The default value is *http://localhost:8080/solr* if this parameter is not defined.
If parameter Solr URLs For Read is not specified, then these urls will be used for reading/writing (advanced queries/indexation).

Parameter Solr URLs for Read

Defines the URL or URLs (valid character separators: comma, semicolon and blank) where search engines Solr are availables. The first Solr identified by its URL will work as master in queries.
The default value is the same of "Parameter Solr URLs", if this parameter is not defined.
With both parameters it is possible to use one server only for queries and another for indexation (write operation). This is useful for Solr cluster installations.

Parameter Solr Indexing Files

Enable or not if the indexation process will manage the content of files with Apache Tika tool. For this purpose, you must define the key "solr_indexing_files_enabled" with these possible values: "true" and "false".

The default value is disabled, if this parameter is not defined.

Parameter Dynamic Loading Folders

Define the folder or folders (valid character separators: comma, semicolon and blank) where the jar files with customized processes are placed.
The default value is *$JBOSS_HOME/dynamic* if this parameter is not defined in `web.xml` file. If you change this folder, you have to copy the *om-sr-cp-#.#.jar* into new location in order to ensure the right performance of customized indexing process.

Parameter Solr Highlight Fields

Optional field that defines the solr fields to highlight for searchs (valid character separators: comma, semicolon and blank) If blank, applies the default value.
Default value is *"full"*. Others valid values are: *customercode, productcode, version …*
More configuration of highlighter can be found on Solr configuration file.

Parameter Validation

Optional field that defines the type of validation those subscriptions are forced to pass against the defined specification. The valid values are: "*STRONG*" (default) and "*SLIGHT*".

- *STRONG*: It is the default type. All the subscription characteristcs and components must validate against the specification definition.
- *SLIGHT*: Subscription characteristics that are defined in the specification follow all the validation rules, but the ones that are not defined can skip this validation but they are not indexed. Child components that are not defined in the specification can skip also this validation, they are indexed anyways.

Parameter Indexation Level

Optional field that defines the detail level to index subscription data. The valid values are: "*LEVEL1*" (default), "*LEVEL2*", "*LEVEL3*".
Three levels for characteristics indexation will be provided:

- *LEVEL1:* This mode will be the default indexation mode and is the current mode, only characteristicName_type pattern is used.
- *LEVEL2*: The componentName_characteristicName_type pattern and LEVEL1's pattern are used.
- *LEVEL3*: The componentName_componentVersion_characteristicName_type pattern and LEVEL2's patterns are used.

Parameter Indexation Mode

Is an optional field that defines indexation mode for indexation levels 2 and 3. Valid values are: "*INCLUSIVE*" (default) and "*EXCLUSIVE*".
Three levels for characteristics indexation mode are provided:

- *INCLUSIVE*: Characteristics are cached in the minimum level (1) and indexed in all the levels under the selected in the property "indexation_level". For example, if LEVEL3 is set, characteristics are cached in level 1 and indexed in levels 1, 2 and 3.
- *EXCLUSIVE*: Characteristics are cached and indexed only in the level set in the property "indexation_level". For example, if LEVEL3 is set, characteristics are cached in level 3 and indexed in level 3 also.

Parameter Security Authorization Status

Is an optional field that defines if user authorization is enable. Default value is 'false', and it is very important set it to 'true' when enabling secure app.
Please read "Securing SR' section.

Parameter system sql console

To use indexationUnplugged process you need to have an oracle sql console like sqlplus installed. During SR installation, installer saves sql console used path (Mysql or Oracle) in a web.xml parameter called "sql_exec_path". When SR starts, it loads this information in a context.

Parameter temp folder

To use indexationUnplugged process you need to specify a directory for build temp files.
The default value is *`$JBOSS_HOME`*/*tmp*.

Parameter all_data_specification_index

To specify that indexation of specification uses all composite information to index (characteristics information). Default value is 'false'.

Parameter history_system_active

Specify that SR saves historic data of subscriptions. If it is true, each update or delete operation generates one entry in subscription_hist table. Default value is false. (If it is false, there are subscription_hist entries only for update solr index (indexing purposes)).

Parameter solr_specifications_indexing_scheduled

Specify specification indexation trigger expression for quartz process.

Parameter indexation_batch_size

Specify the size of subscription batch to index. Default value is 40000 (on each iteration of indexation process, 40000 subscriptions are retrieved from database and send to Solr, in only one transaction).

Parameter enable_related_subscriptions_indexation

Specify whether it is desired to handle subscription dependencies. If it is not defined it is understood default value false, in case of activation required it must be added and set to true in DB manually.

5.2.3. Solr Configuration File

The Solr configuration is a task done by configuration process and it does not require any manual changes. There is one file implicated in Solr configuration:

- `$SOLR_HOME/conf/solrconfig.xml`

In the `solrconfig.xml` file all the solr configuration is defined. Normally the unique change required could be modifying the maximum of concurrent requests supported:

```
<query>
  ...
  <useColdSearcher>true</useColdSearcher>
  <maxWarmingSearchers>?max_warming_searchers</maxWarmingSearchers>
</query>
```

The main configurable variable is:

Variable	Description
?max_warming_searchers	Maximum number of searchers that may be warming in the background concurrently. Default value after installation is 100.

For additional information see Solr official site [http://lucene.apache.org/solr/].

5.2.4. Solr Schema Configuration File

Default configuration is ready to use. This paragraph can be used to configure SR advanced searches for having special filters or tokenizers to manage results. For instance, inside of *full* field is saved XML file. Nevertheless, if user does an advanced search by text, results are filtered only by data, not by tags. This behaviour is defined in configuration file:

- `$SOLR_HOME/conf/schema.xml`

In paragraph:

```
<fieldType name="xmlString" class="solr.TextField" omitNorms="true">
        <analyzer>
                <tokenizer class="solr.HTMLStripWhitespaceTokenizerFactory" />
                <filter class="solr.ReversedWildcardFilterFactory" />
```

```
                    <filter class="solr.LowerCaseFilterFactory" />
            </analyzer>
</fieldType>
```

In this case, we have defined a new type of field that filters all saved/searched (index/query) data. Analyzer can be of type: "query" or "index" or none (in this case, it applies both). Inside of analyzer, there are one tokenizer and two filters. HTMLStripWhitespaceTokenizerFactory is used to remove xml tags, and filters are used to search by wildcard and not sensitive case searches.

Remember that these changes only apply for advanced searches, and any change in this file requires a full reindexation.

If you need more information about this configuration, read documentation of Solr [http://wiki.apache.org/solr/AnalyzersTokenizersTokenFilters].

NOTE: Some configuration could be not valid: SR doesn't start, SR starts and runs without errors but has not desire behaviour…
After a change, it´s good idea to test, but user must not use Solr web interface (Only web service interface can reproduce production server enviroments).

5.2.5. Log Configuration File

The log configuration is a task done by configuration process and it doesn't require any manual change. There is one file implicated in log4j configuration:

- `$JBOSS_HOME/server/default/conf/jboss-log4j.xml`

In the `jboss-log4j.xml` file all the log4j configuration is defined. Normally the unique change required could be modifying the log level:

```
<root>
  <priority value="?log_level"/>
  <appender-ref ref="CONSOLE"/>
  <appender-ref ref="FILE"/>
</root>
```

The main configurable variable is:

Variable	Description
?log_level	Level defined for log traces.

For additional information see Log4j official reference [http://logging.apache.org/log4j/].

5.3. Advance configuration

(For detailed information and monitor production servers is available: "*HP Subscription Repository - Tuning & Performance Technical Note.doc*").

The first step to (advance) configure SR is to decide the maximum connection (requests) number will be running concurrently. Logically, hardware always marks the limit. For instance, each oracle session consumption around one megabyte (depends of version and configuration), so to support 200 connections, you should have a disk with at least 200 free megabytes. (Besides, oracle spends more

space with tablespace, memory …) In short, you should configure the system to support maximum connections number. It´s an unnecessary spend uses more connections.

If you establish a connection number too low, system could reject request or block it.

There are three points to configure the connections limit:

1. Database.
2. Application Server.
3. Application.

5.3.1. Database configuration

To check the connection number in an Oracle database, you should execute this query:

```
SELECT name, value
  FROM v$parameter
 WHERE name = 'sessions'
```

And:

```
SELECT name, value
  FROM v$parameter
 WHERE name = 'processes'
```

There are two different things: connection and session. (Please, read database documentation for more information).

To change these values, you should execute this sql:

```
alter system set sessions=<newvalue> scope=spfile;
alter system set processes=<newvalue> scope=spfile;
```

It is necessary to reboot the system to apply the changes.

NOTE (Remember that database could be allocate more services, so you should calculate for all databases that are running in the same Oracle).

Oracle configuration can be modified to get better performance with big subscription size requests, please notice that these changes modify the general performance so, It is recommended to manage them under the supervision of a DBA. Possible suggested changes in CLOB fields:

- Disable cache
- Disable logging
- Disable storage in row
- Modify the lob extension parameters

5.3.2. Application server configuration

To improve performance in an application server is necessary modify:

- Log level detail (more logs is equal than more disk writes).
- Memory configuration (you should use only necessary memory. JVM runs slow if has a lot of memory to manage. And application crashs if hasn't memory.)
- Establish the max number of threads running concurrently.

Normally, in production systems is good idea remove unused applications or addons of Jboss. Because users or administrators don't use frequently and burn resources as cpu or memory. (Please read Jboss Admin documentation for more information).

To configure log, you should open the file:

- `..\jboss\server\default\conf\jboss-log4j.xml`

You could comment "<category>" elements to restrict that packages will be logged and configure log level. By default, in development environment it´s normal to use "INFO" log level to view all information. In production, you should use a high log level, like "ERROR" (Please, read log4j documentation for more information about log levels). So, file writes will be restricted, and performance will be better.

Remember that it´s best solution remove log entry that increase log level.

To configure memory you should modify:

- `..\jboss\bin\run.conf.bat`

There are two basic entries to manage memory:

```
set "JAVA_OPTS=-Xms256M -Xmx4072M -XX:MaxPermSize=128M"

set "JAVA_OPTS=%JAVA_OPTS% -Dsun.rmi.dgc.client.gcInterval=3600000 -Dsun.rmi.dgc.server.gcInterval=3600000"
```

Memory configuration is established with first entry. You can add more configuration parameters, but you should configure initial memory, max memory and object generation memory at least. Above configuration line has been defined for a server with 8 Gb of memory, where there are running a Oracle and Jboss server.

Except application with memory cache (where cache (and memory) can increase in execution time), memory consumption should be lineal. It´s good idea to start with more reserved memory, and decrease/adjust to production consumption data. Remember that JVM spends time to manage memory, because it´s important monitor application comsumptions during first days.

Normally it´s unnecessary (and bad idea) to modify garbage collector configuration (second line). Garbage collector manages itself automatically. Specific configurations are best for exceptional cases.

To configure maximum number of listener threads, modify file:

- `..\jboss\server\default\deploy\jbosssweb.sar\server.xml`

And add "maxThreads" parameter to listen connector (normally it´s 8080 port connector or 443/8443 secure port connector). For example:

```
<Connector protocol="HTTP/1.1" port="8080" address="${jboss.bind.address}"
connectionTimeout="20000" redirectPort="8443" URIEncoding="UTF-8"  maxThreads="150"/>
```

Like database server, it´s important to have enough threads for ALL (not only SR) application running in Jboss. Normally, depends of software architecture, each request burn at least one thread. SR burns two threads per request, one for user requet and another for SR to Solr request.

5.3.3. Application configuration

At last step, basic application configuration is required. Depends of customers, we could use several parameters: scheduled indexation … (Please read SR manual for more information).

In other sections we cover how adapt database configuration to use all oracle available connections. Now, we must specify in SR datasource configuration that maximize its connection pool.

For this, edit file:

- `..\jboss\server\default\deploy\subscriptionrepository-ds.xml`

And adapt pool size and idle timeouts. If you need that application responses 150 clients and oracle database offers 150 connections, you must establish a maximum pool size at least of 150. In this case, this pool is used only by SR. In the pool there are available jdbc connections, and when a connection is released go back to pool. (In this step, we use jdbc connections).

Besides, it´s good idea to decrease timeout times to release "forgotten" connections. This is a basic configuration:

```
<min-pool-size>5</min-pool-size>
<max-pool-size>150</max-pool-size>
<blocking-timeout-millis>10000</blocking-timeout-millis>
<idle-timeout-minutes>1</idle-timeout-minutes>
```

Normally, application uses 3 – 5 connections to load cache and other maintance operations. Besides, Indexation or migration process uses more connections too. Because of this, you should add SR connections plus process connections plus users connections, to estimate best approach. (In all cases, if min-pool-size is 100, then jboss opens 100 connections when starts. This is a overload of network traffic, but it could improve connection time (and consequently all times) for slow databases.)

5.3.4. Jboss cluster configuration

To solve cluster configuration problems, SR uses a centralized configuration, join all "cluster information" in a database. Normally, if SR has been installed with Sr installer, the configuration has been set correctly.

For instance, if you deside to have three servers to maximize performance; two for Jboss and one for Oracle, you should decide what will be master node. (A master node will launch scheduled indexation and optimization process). If you have only one server, then it will be master node. To configure this, installer introduces in web.xml file a param called "master_node", whose value will be true or false (only for slave nodes, when there are more than one Jboss server).

NOTE: SR supports several master nodes to let high availibity environment configuration. In this case, all master nodes will try to launch process, but only first can do. Syncronization between nodes is done in database. SR architecture limits concurrently access for process, only in this special case.

Master node configuration is only in web.xml because is not share between nodes. File example:

```
<web-app>

        <display-name>HP Subscription Repository</display-name>

        <servlet>
                <servlet-name>subscription_repository</servlet-name>
                <servlet-class>com.hp.om.sr.servlet.SRServlet</servlet-class>
                <init-param>
                        <param-name>inventory_type</param-name>
                        <param-value>DB_ORACLE</param-value>
                </init-param>
                <init-param>
                        <param-name>max_number_results</param-name>
                        <param-value>100</param-value>
                </init-param>
                <init-param>
                        <param-name>sql_exec_path</param-name>
                        <param-value>sqlplus</param-value>
                </init-param>
                <init-param>
                        <param-name>master_node</param-name>
                        <param-value>true</param-value>
                </init-param>

                <load-on-startup>1</load-on-startup>
        </servlet>
...
```

In a previous example, one node will be master (master_node=true) and other slave (master_node=false). If you configure both as master can produce more log errors because SR detects both indexation process running concurrently and stops last process launched. If you configure both as slave and indexation is scheduled, SR doesn´t index subscriptions and queries can return bad responses. So it´s best solution to configure both as master if you have doubts.

Next steps are to configure solr and sr urls in database. If you desire to do manually, you have to insert a row by each node. For previous example, with two jboss servers and sr servers:

```
Insert into CONFIGURATION (KEY,VALUE) values ('sr_urls','http://16.19.142.141:8080/subscriptionrepository');
Insert into CONFIGURATION (KEY,VALUE) values ('sr_urls','http://16.19.141.72:8080/subscriptionrepository');
Insert into CONFIGURATION (KEY,VALUE) values ('solr_urls','http://16.19.141.72:8080/solr');
Insert into CONFIGURATION (KEY,VALUE) values ('solr_urls','http://16.19.142.141:8080/solr');
```

(NOTE: Key column has not unique restriction).

Optionally, you can add more cluster information like:

```
Insert into CONFIGURATION (KEY,VALUE) values ('solr_indexing_state','true');  -- SR variable, don´t modify manually.
Insert into CONFIGURATION (KEY,VALUE) values ('max_number_results','10');
Insert into CONFIGURATION (KEY,VALUE) values ('indexation_level','LEVEL1');
Insert into CONFIGURATION (KEY,VALUE) values ('solr_indexing_scheduled','0 30 23 ? * SAT');
Insert into CONFIGURATION (KEY,VALUE) values ('solr_optimization_scheduled','0 30 23 ? * SAT');
```

Note that this configuration only can exist one time in database. You must review Configuration table and check that only exist one "max_number_results" entry. If you use Sr installer, don´t have problems because this problem it´s managed by installer.

This previous information is loaded during SR starting, so you should reboot SR´s when modify database configuration to apply changes. If any mandatory "cluster" configuration is missing, SR launches error during start.

5.4. Running SR with other services

5.4.1. Conflicts with ports

If you are running SR with other services that are running with Jboss or Apache Tomcat, you could have problems with ports. By default, all web application servers will try to start at port: 8080. So, first server (in starting sequence) reserves needed ports; and anothers cannot start.

To solve this problem, administrator must complete two steps. First, he should configure SR Jboss to use other ports. Edit file:

- `..\jboss\bin\run.conf.bat`

And add this text "-Djboss.service.binding.set=ports-01" in the line:

```
set "JAVA_OPTS=%JAVA_OPTS%
```

Line must be:

```
set "JAVA_OPTS=%JAVA_OPTS% -Djboss.service.binding.set=ports-01"
```

Remember that this line could contain other jboss options. Additional information about this jboss option can be founded in: http://docs.redhat.com/docs/en-US/JBoss_Enterprise_Web_Platform/5/html/Getting_Started_Guide/The_Service_Binding_Manager.html

Second, user must configure Solr port in SR configuration (SR and Solr had started in port 8180, but SR (by default) try to connect with Solr using 8080 port). For this purpose, there are two options:

1. Single Node configuration.

   Modify file:

```
.. \jboss\server\default\deploy\subscriptionrepository.war\WEB-INF\web.xml
```

   and add parameter "solr_urls". For instance:

```
<servlet>
…
        <init-param>
                <param-name>solr_urls</param-name>
                <param-value>http://localhost:8180/solr</param-value>
        </init-param>
…
```

2. Cluster configuration.

   Administrator has to update/insert row into "CONFIGURATION" table (of SR tablespace).

```
Insert into CONFIGURATION(key,value) values ('solr_urls','http://localhost:8180/solr');
```

In this row, there are all solr ip addresses of cluster. It should modify port of desire servers. If both configurations are defined, SR will use database configuration.

6. Security and authorization

This section covers the security configuration.

6.1. Secure transport

There are two steps to define that SR manages encrypted data through SSL channel:

Create keystore to save our certificates. Keystore file should be located in one directory where SR has read privileges. Normally, it´s good idea to backup keystore file. (If you have one of previous installation, you can use it (and pass this step), but you must have the password.)

To create keystore, run command line and go to jdk/bin directory. For example:

- cd C:\java\jdk1.6.0_24\bin

or

- cd %JAVA_HOME%\bin

Then execute keystore (and key) creation command:

- keytool -genkey -alias jbossws -keyalg RSA -keystore "%JBOSS_HOME%\conf\SR.keystore"

You should complete the next steps:

```
Escriba la contrase±a del almacÚn de claves:
Volver a escribir la contrase±a nueva:
¬Cußles son su nombre y su apellido?
  [Unknown]:  HP
¬Cußl es el nombre de su unidad de organizaci¾n?
  [Unknown]:  Delivery
¬Cußl es el nombre de su organizaci¾n?
  [Unknown]:  HP
¬Cußl es el nombre de su ciudad o localidad?
  [Unknown]:  Las Rozas
¬Cußl es el nombre de su estado o provincia?
  [Unknown]:  Madrid
¬Cußl es el c¾digo de paÝs de dos letras de la unidad?
  [Unknown]:  es
¬Es correcto CN=HP, OU=Delivery, O=HP, L=Las Rozas, ST=Madrid, C=es?
  [no]:  si

Escriba la contrase±a clave para <jbossws>
        (INTRO si es la misma contrase±a que la del almacÚn de claves):
Volver a escribir la contrase±a nueva:
```

The first step requests a keystore (file access) password. (Remember this password, you should use in server.xml configuration). The last step requests other password to encrypt data.

(This is common way to create keystores with java utilities, but exists more). It can exist more keystore (file), but only one for each Jboss-Tomcat. Also it can exist several keys in the same keystore (file). Normally, if exist only one application in each jboss server, it should exist one keystore for each server.

The second step is to configure the application for use the keystore, you should open tomcat server configuration file:

- $JBOSS_HOME/server/default/deploy/jbossweb.sar/server.xml

You have to change the address parameter of standard connector (from "*${jboss.bind.address}*" to "*127.0.0.1*"), and change *redirectPort* to same port that you establish in secure connector (normally "443"). In this way, you configure jboss to listen only local requests, so local solr request will be processed more faster without encryption. If you don't change address parameter, then request can be receipt in two ports 8080 and secure port:

```
<!-- A HTTP/1.1 Connector on port 8080 -->
<Connector protocol="HTTP/1.1" port="8080" address="127.0.0.1"
        connectionTimeout="20000" redirectPort="443" URIEncoding="UTF-8" />
```

It´s good idea that you comment old lines, and work with copied lines. So you can do rollback delete copied lines.

After you should enable (clear comment simbols) the secure connector:

```
<!-- A AJP 1.3 Connector on port 8009 -->
                        <Connector protocol="HTTP/1.1" SSLEnabled="true"
                                port="443" address="${jboss.bind.address}"
                                scheme="https" secure="true" clientAuth="false"
                                keystoreFile="${jboss.server.home.dir}/conf/SR.keystore"
                                keystorePass="jbossws"
                                truststoreFile="${jboss.server.home.dir}/conf/SR.keystore"
                                truststorePass="jbossws"
                                sslProtocol = "TLS" />
```

To configure the connector, it´s very important that:

- *port* : it´s the listen port to app. For example, if it´s 443, then you should use: "https://localhost/subscr… ". If it´s 8443, then you should use: "https://localhost:8443/subscr …" and so on. (If you use por 443, the url will be "https://localhost/"). This port must be the same that "*redirectPort*" in others connectors.
- *keystoreFile* and *truststoreFile*: contain the keystore file path (remember the path used in previous step).
- *keystorePass* and *truststorePass*: contain the keystore password (remember the first password used in previous step).

After you should reboot Jboss server for apply new configuration. At this point, the basic URL http://localhost:8080/ converts to https:/localhost/ (or https://localhost:8443/). Remember this when you want invoke SR or Solr.

With this section we have cover data encryption. Somebody can access to web service and its methods. To restrict access, read next section.

(Pay attention if you configure in "web.xml" file to set "<transport-guarantee>", because all auth requests are sent again secure port (normally 443). So, you should enable secure connector. Besides, if you want forbid all connections by 8080 port (localhost included), you must configure solr to use secure http client. Please, read Developer Guide.doc about this).

6.2. Authorization

This section covers the configuration to restrict access.

There are several roles for each End Point method:

OperationsEndPoint	*ManagementEndPoint*
createSubscriptionRole	cleanHistoricProcessesRole
updateSubscriptionRole	finishProcessRole
deleteSubscriptionRole	processStatusRole
querySubscriptionRole	allProcessStatusRole
advancedQuerySubscriptionRole	executeProcessByNameRole
createSpecificationRole	executeProcessRole
updateSpecificationRole	executeIndexProcessRole
deleteSpecificationRole	executeReindexProcessRole
querySpecificationRole	listAvailableProcessesRole
deleteAdvancedQueryConfRole	SRStatusRole
listAdvancedQueryConfRole	getConfigurationParameterRole
getAdvancedQueryConfRole	reloadCacheRole
saveAdvancedQueryConfRole	setConfigurationParameterRole
advancedQuerySpecificationRole	executeSpecificationIndexProcessRole
deleteSpecificationByQueryRole	executeSpecificationReindexProcessRole
getAttachmentQueryRole	executeSubscriptionIndexProcessRole
deleteAttachmentRole	executeSubscriptionReindexProcessRole
deleteBasicSubscriptionElementRole	
createBasicSubscriptionElementRole	
updateBasicSubscriptionElementRole	
queryBasicSubscriptionElementRole	
advancedQueryBasicSubscriptionElementRole	
createDependenciesRole	
deleteDependenciesRole	
updateUsedDependenciesRole	
getSourceDependenciesRole	
getTargetDependenciesRole	
requestGetterSpecificationPropertiesRole	
requestSetterSpecificationPropertiesRole	
requestGetterSubscriptionPropertiesRole	
requestSetterSubscriptionPropertiesRole	

1. The first step configures the application as auth required application. To do this, you should open webapp config file:
   - %JBOSS_HOME%/server/default/deploy/subscriptionrepository.war/WEB-INF/web.xml

   (This file is defined in development environment).

   Add the next paragraph at the end of file (it´s mandatory maintain the params order in the file):

```
…
        <security-constraint>
```

```
                <web-resource-collection>
                        <web-resource-name>App Security</web-resource-name>
                        <url-pattern>/*</url-pattern>
                        <http-method>POST</http-method>
                        <!-- If you wish restrict access to wsdl file, add http-method: GET -->
                </web-resource-collection>
                <auth-constraint>
                        <role-name>appRole</role-name>
                </auth-constraint>
        </security-constraint>

        <login-config>
                <auth-method>BASIC</auth-method>
                <realm-name>SR Realm</realm-name>
        </login-config>
...
```

If you desire to use hashed password in configuration file, you have to replace auth-method: "BASIC" and put: "DIGEST".

2. Next step configures the server to use security. To do this, you should open webapp config file:
   - %JBOSS_HOME%/server/default/deploy/subscriptionrepository.war/WEB-INF/jboss-web.xml

   (This file is defined in development environment).

   Add the next paragraph at the end of file (it´s mandatory maintain the params order in the file):

```
...
<security-domain>java:/jaas/JBossWS</security-domain>
...
```

3. Configure the application to authorize. To do this, you should open webapp config file:
   - %JBOSS_HOME%/server/default/deploy/subscriptionrepository.war/WEB-INF/web.xml

   (This file is defined in development environment).

   Add the parameter: "*security_authorize_status*":

```
...
        <servlet>
                <servlet-name>subscription_repository</servlet-name>
                <servlet-class>com.hp.om.sr.servlet.SRServlet</servlet-class>
                <init-param>
                        <param-name>inventory_type</param-name>
                        <param-value><!-- Template:OM_INVENTORY_TYPE --></param-value>
                </init-param>
                <init-param>
                        <param-name>max_number_results</param-name>
                        <param-value>100</param-value>
                </init-param>
                <init-param>
                        <param-name>security_authorize_status</param-name>
                        <param-value>true</param-value>
                </init-param>
...
```

4. Only if you use digested passwords, you have to modify the file:
    - %JBOSS_HOME%/server/default/conf/login-config.xml

   And replace *UsersRolesLoginModule* configuration. This is the old:

```
<application-policy name="JBossWS">
  <authentication>
    <login-module code="org.jboss.security.auth.spi.UsersRolesLoginModule"
      flag="required">
      <module-option name="usersProperties">props/jbossws-users.properties</module-option>
      <module-option name="rolesProperties">props/jbossws-roles.properties</module-option>
      <module-option name="unauthenticatedIdentity">anonymous</module-option>
    </login-module>
  </authentication>
</application-policy>
```

   New configuration will be:

```
<application-policy name="JBossWS">
  <authentication>
    <login-module code="org.jboss.security.auth.spi.UsersRolesLoginModule"
      flag="required">
      <module-option name="usersProperties">props/jbossws-users.properties</module-option>
      <module-option name="rolesProperties">props/jbossws-roles.properties</module-option>
      <module-option name="unauthenticatedIdentity">anonymous</module-option>
                    <module-option name="hashAlgorithm">MD5</module-option>
      <module-option name="hashEncoding">rfc2617</module-option>
      <module-option name="hashUserPassword">false</module-option>
      <module-option name="hashStorePassword">true</module-option>
      <module-option name="passwordIsA1Hash">true</module-option>
      <module-option name="storeDigestCallback">org.jboss.security.auth.spi.RFC2617Digest</module-option>
    </login-module>
  </authentication>
</application-policy>
```

5. The last step it´s doing user-password-roles configurations. You should locate two files:
    - %JBOSS_HOME%/server/default/conf/props/jbossws-users.properties
    - %JBOSS_HOME%/server/default/conf/props/jbossws-roles.properties

   In the first file you can define pairs of user=password. For instance:

   Myuser=mySecretPassword

   or

   (Digested example) Myuser=56726a2f5bbde5339a7cbe8a464a3a60

   In the second file, you should define associated roles to existing users. For instance:

   Myuser=appRole,SRStatusRole

Each user must have "appRole" (at least), to can connect with application. Besides, you should add desired roles (using commas as separator). We have grouped roles by end point:

```
#Only Operations roles formatted
#createSubscriptionRole,updateSubscriptionRole,deleteSubscriptionRole,querySubscriptionRole,advancedQuerySubscriptionRole,createSpecificationRole,updateSpecificationRole,deleteSpecificationRole,querySpecificationRole,deleteAdvancedQueryConfRole,listAdvancedQueryConfRole,getAdvancedQueryConfRole,saveAdvancedQueryConfRole, advancedQuerySpecificationRole, deleteSpecificationByQueryRole

#Onlye Management roles formatted
#cleanHistoricProcessesRole,finishProcessRole,processStatusRole,allProcessStatusRole,executeProcessByNameRole,executeProcessRole,executeIndexProcessRole,executeReindexProcessRole,listAvailableProcessesRole,SRStatusRole, getConfigurationParameterRole, reloadCacheRole, setConfigurationParameterRole, executeSpecificationIndexProcessRole, executeSpecificationReindexProcessRole, executeSubscriptionIndexProcessRole, executeSubscriptionReindexProcessRole

#ALL ROLES
#cleanHistoricProcessesRole,finishProcessRole,processStatusRole,allProcessStatusRole,executeProcessByNameRole,executeProcessRole,executeIndexProcessRole,executeReindexProcessRole,listAvailableProcessesRole,SRStatusRole,createSubscriptionRole,updateSubscriptionRole,deleteSubscriptionRole,querySubscriptionRole,advancedQuerySubscriptionRole,createSpecificationRole,updateSpecificationRole,deleteSpecificationRole,querySpecificationRole,deleteAdvancedQueryConfRole,listAdvancedQueryConfRole,getAdvancedQueryConfRole,saveAdvancedQueryConfRole, getConfigurationParameterRole, reloadCacheRole, setConfigurationParameterRole, advancedQuerySpecificationRole, executeSpecificationIndexProcessRole, executeSpecificationReindexProcessRole, executeSubscriptionIndexProcessRole, executeSubscriptionReindexProcessRole, deleteSpecificationByQueryRole
```

Alternatives locations for jbossws-…properties files can be specified in: %JBOSS_HOME%/server/default/conf/login-config.xml

(Only for digested passwords) To generate hashed password you have to execute:

```
cd %JBOSS_HOME%/server/common/lib
java –cp jbosssx-server.jar org.jboss.security.auth.spi.RFC2617Digest USERLOGIN “SR Realm” USERPASSWORD
```

where USERLOGIN and USERPASSWORD must be user data. System generates one text like this:

```
RFC2617 A1 hash: 56726a2f5bbde5339a7cbe8a464a3a60
```

Hashed password is: “56726a2f5bbde5339a7cbe8a464a3a60”

It is not recommended to use authentication and authorization without SSL encryption, because passwords are sent as text. It´s more secure define “DIGEST”, in “*<auth-method>*” of web.xml, to use hashed password. But hashed encryption methods can be broken.

Besides, it can use other users-roles management police: database system (users, passwords and roles will be saved in database). Normally, this configuration is used in webapp enviroments, but it´s not used in web service due management costs). Please read Jboss documentation for more information.

If you want only restrict access to ALL application, but not each method; you must set servlet param “*security_authorize_status*” to false (previous section 3) and use only “appRole”. In this case, users can access or not to all web service methods.

If you have two available connections types, one port secure (SSL) and another in a insecure port, you can restrict all communications with auth information (user and password) to use secure connection. For this, you must specify within a <security-constraint> element (in WEB-INF/web.xml file):

```
<user-data-constraint>
  <description>Require SSL</description>
  <transport-guarantee>CONFIDENTIAL</transport-guarantee>
```

```
</user-data-constraint>
```

6.3. Limit by IP address

It´s possible restrict the access to certain ip´s. There are several ways to do: Tomcat configuration … but the less intrusive way is doing configuration in web.xml (only affects SR).

You should open %JBOSS_HOME%/server/default/deploy/subscriptionrepository.war/WEB-INF/web.xml and add the next paragraph (be careful with order):

```
<web-app>

        <display-name>HP Subscription Repository</display-name>

         <filter>
                <filter-name>RemoteHostFilter</filter-name>
                <filter-class>org.jboss.remotehostfilter.RemoteHostFilter</filter-class>
                <init-param>
                        <param-name>allow</param-name>
                        <param-value>16.19.142.*</param-value>
                </init-param>
         </filter>

         <filter-mapping>
                <filter-name>RemoteHostFilter</filter-name>
                <url-pattern>/*</url-pattern>
         </filter-mapping>
…
```

There are two parts: filter declaration and filter mapping. In first part you define: name and params of filter, and in the second: url affected. It can be restrict access only to Management operations (*"/management/*"*), for example; or all application, like this text.
You can use a ip address or mask: 192.168.*.*, 192.168.1.* … and use comma character to separate ip´s. In the example, only use "allow" param; but it´s possible use more.

Filter operation:

- If there are any "deny" expressions configured, the IP will be compared to each expression. If a match is found, this request will be rejected with a "Forbidden" HTTP response.
- If there are any allow expressions configured, the IP will be compared to each such expression. If a match is NOT found, this request will be rejected with a "Forbidden" HTTP response.
- Otherwise, the request will continue normally.

6.4. Secure web.xml file (example).

```
<?xml version="1.0" encoding="utf-8"?>
        <!--
                Copyright (c) 2000-2003 Hewlett-Packard Company. All Rights Reserved
```

```
        ->

<!DOCTYPE web-app
  PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
  "http://java.sun.com/dtd/web-app_2_3.dtd">

<web-app>

        <display-name>HP Subscription Repository</display-name>

         <filter>
                <filter-name>RemoteHostFilter</filter-name>
                <filter-class>org.jboss.remotehostfilter.RemoteHostFilter</filter-class>
                <init-param>
                  <param-name>allow</param-name>
                        <param-value>16.19.*.*, 127.0.0.1</param-value>
                </init-param>
          </filter>

          <filter-mapping>
                <filter-name>RemoteHostFilter</filter-name>
                <url-pattern>/*</url-pattern>
          </filter-mapping>

        <servlet>
                <servlet-name>subscription_repository</servlet-name>
                <servlet-class>com.hp.om.sr.servlet.SRServlet</servlet-class>
                <init-param>
                        <param-name>inventory_type</param-name>
                        <param-value>DB_ORACLE</param-value>
                </init-param>
                <init-param>
                        <param-name>max_number_results</param-name>
                        <param-value>100</param-value>
                </init-param>
                <init-param>
                        <param-name>security_authorize_status</param-name>
                        <param-value>true</param-value>
                </init-param>
                <load-on-startup>1</load-on-startup>
        </servlet>

        <servlet>
                <servlet-name>subscription_repository_operations_endpoint</servlet-name>
                <servlet-class>com.hp.om.sr.endpoint.OperationsEndPoint</servlet-class>
                <load-on-startup>2</load-on-startup>
        </servlet>


        <servlet>
                <servlet-name>subscription_repository_management</servlet-name>
                <servlet-class>com.hp.om.sr.endpoint.ManagementEndPoint</servlet-class>
                <load-on-startup>3</load-on-startup>
        </servlet>


        <servlet-mapping>
                <servlet-name>subscription_repository_operations_endpoint</servlet-name>
                <url-pattern>/operations/*</url-pattern>
        </servlet-mapping>


        <servlet-mapping>
```

```
                <servlet-name>subscription_repository_management</servlet-name>
                <url-pattern>/management/*</url-pattern>
        </servlet-mapping>


        <resource-ref>
                <res-ref-name>DefaultSR1</res-ref-name>
                <res-type>javax.sql.DataSource</res-type>
                <res-auth>Container</res-auth>
        </resource-ref>


        <security-constraint>
                <web-resource-collection>
                        <web-resource-name>App Security</web-resource-name>
                        <url-pattern>/*</url-pattern>
                        <http-method>POST</http-method>
                        <!-- If you wish restrict access to wsdl file, add http-method: GET -->
                </web-resource-collection>
                <auth-constraint>
                        <role-name>appRole</role-name>
                </auth-constraint>
        </security-constraint>

        <login-config>
                <auth-method>BASIC</auth-method>
                <realm-name>SR Realm</realm-name>
        </login-config>
</web-app>
```

7. Troubleshooting

This section describes some problems and their possible solutions.

7.1. Inconsistences between Database and Index

In case inconsistences between database and index are detected, there are two solutions taking into account if you can found clearly the inconsistences in a little set of registers (minor inconsistences) or not (severe inconsistences). Obviously if you are using scheduled index, the differences derived from changes from the last indexation aren't considered as inconsistences.

7.1.1. Minor Inconsistences

If the inconsistences are located in a little set of registers you can solve the problem doing some changes in database:

- If a register of database has wrong indexed values, mark the field indexed as 'false' in subscriptions table and execute the index process.
- If a register of database doesn't appear in index, ensure that the field indexed in subscriptions table has value 'false' and execute the index process.
- If a register is in index but it doesn't exist in database, add a register in table subscriptions_hist using the id of register in index as id of new register in this table, and execute the index process.

7.1.2. Severe Inconsistences

If the inconsistences aren't located in a little set of registers, a reindexation is probably necessary. This reindex process is really critical, because is a process that could deteriorate the performance of system and the duration is approximately 10 millions every day and a half. Due to these previous considerations in the next reindexation plan is proposed to use a pre-production environment:

- Install the system in the pre-production environment (see 'Installation' chapter) and use the production DBMS as datasource.
- Fix in pre-production environment the value of parameter 'Search Engine Reindexing Mock Mode' as 'true' (see 'Application Configuration File' in previous chapter). This is a step very very important.
- Stop the production environment if it is working with online indexing and schedule the indexing (see 'Application Configuration File') for a far date in future (i.e. one month or one year in future).
- Start again the production environment.
- Start the pre-production system and invoke the reindex process. This process reindex approximately 10 millions every day and a half or two days.
- After accomplish successfully reindexation stop production and pre-production environments.
- Remove the Solr data folders of production and overwrite only the data folders generated in pre-production into production environment. Only Solr data folders are removed and overwritten in this step.
- If production environment was working in online indexing mode, then put again the indexing in online mode.
- Start again the production environment and invoke the index process in order to propagate the changes done from the begin of reindex plan into reindexed index.

7.2. OutOfMemory Exception

This kind of exception could be due to the RAM memory assigned is insufficient. Stop the system and increase the RAM memory assigned to JVM.

There are two files for increasing memory. In windows:

- %JBOSS_HOME%/bin/run.conf.bat

In Linux/Unix:

- %JBOSS_HOME%/bin/run.conf

Although there are several memory parameters, only we manage three:

- Xms: Initial heap size
- Xmx: Max. heap size
- XX:MaxPermSize : Size of the Permanent Generation

Default configuration (for Windows) is:

```
set "JAVA_OPTS=-Xms128M -Xmx800M -XX:MaxPermSize=256M"
```

Normally, java exception trace logs shows memory zone where it´s problem: heap size problem, perm. gen… Please read Jboss documentation for more information if administrator has problems after increasing *Xmx* and *XX:MaxPermSize* parameters.

7.3. Continuous Performance Decrease

If a continuous performance decrease is detected, is recommended use scheduled indexing mode instead of online indexing mode, modifying the parameter 'Solr Indexing Scheduled' (see subsection 'Application Configuration File' in chapter 'Configuration'), if after this improvement the problem persists, then increase the nodes in cluster installation (see section 'Installation Solr Cluster' in chapter 'Installation').

7.4. Poor Performance

See proposed solutions for 'Continuous Performance Decrease'.

7.5. Process scheduling

To evict process overlapping and data damage, SR forbids run two process of same type concurrently. This can produce error log traces, but really it is not error.

For instance, if you plan indexation process to launch each five minutes, but each indexation spends twenty minutes; then three last indexation processes are not launched because first is running. Besides, log registers traces where SR specifies that three processes cannot be launched.

Admin should re-schedule indexation.

8. General Performance Recommendations

8.1. General configuration

There are some generic advises:

- Set log level to Error level (Log Configuration File) (Restrict hard disk operations)
- Configure Jboss max. connections and Oracle pool size (Database configuration, Application configuration, Application server configuration) (Improve concurrent connections)
- Configure Jboss memory: 4 Gb of memory it´s usual configuration (Application server configuration) (Improve general performance of Jboss)

With this basic configuration the system should run correctly. If any configuration parameter of last paragraph was incorrect, then system doesn´t start. If during normal function server throws exception related with memory problems, a common solution is increase memory in configuration.

Memory consumption should be lineal, but depends of some processes that could use all memory. For example, indexation process with a batch size too long.

8.2. Subscription Repository configuration

8.2.1. Indexation

Next point to improve performance is SR configuration. Indexation process is a key piece. User has two options: offline or online indexation (you can choose any of them during installation or configuration process). Normally is best option uses online indexation.

Online indexation benefits

- It´s unnecessary indexation process.
- Slow creation or update or delete operations.
- Data is syncronized between database and solr (Advanced queries return results in real time).

Offline indexation benefits

- Quick creation or update or delete operations.
- Data is not syncronized between database and solr (Advanced queries do not return results in real time).
- It´s necessary indexation process (better general performance).

Normally, choice depends of user requirements. If user requires a low response time but has inactive periods, then is better to use OfflineIndexation. During inactive periods can run indexation process. Next data (of 100 concurrent requests) can be used to decide the best configuration.

For insert:

- Average time with Online Indexation: 980 ms.
- Average time with Offline Indexation: 16 ms.

For update:

- Average time with Online Indexation: 602 ms.
- Average time with Offline Indexation: 38 ms.

For delete:

- Average time with Online Indexation: 996 ms.

- Average time with Offline Indexation: 18 ms.

(These times has been recorded for a server with 4 Xeon processors and 8 gb of RAM memory, with a empty database. This server has been used for all examples).

In other side, there are indexation process times. With two processes running concurrently (one for insertion and another for indexing):

Subscriptions inserted	***Subscriptions indexed***
0 subs/second	700 subs/second
200 subs/second	555 subs/second
638 subs/second	200 subs/second

Besides, if SR is inserting 200 subs/second, then indexation process spends three minutes for 100000 subscriptions. If insertion speed is 638 subs/second, then indexation time for 100000 subscriptions was eight minutes. So we can establish a limit for 200-300 subscriptions per second. If user uses offline indexation and inserts more than 200-300 subs/second, then indexation process never can index all subscriptions (concurrently with insertion/update operations).

This data depends of server performance and size of database. With a minimal data and without user request, indexation process can index 850 subscriptions per second.

If SR is configured to use offline indexation, the best balance is schedule Indexation process to be launched during activity free windows (without user requests). On the night, for example (the big "but" is that could exist data not syncronized).

Indexation process has a configuration value to define subscriptions batch size (Application Configuration File). This is readed, from database configuration table, when the indexation process starts (can change it without reboot). Default size is 40000 subscriptions. In this case, the balance is between memory comsuption and database/solr requests. If user establishes a batch size too long, then SR can crash with OutOfMemoryException. But if user defines a batch size too short, then SR does a lot of queries to database and Solr.

Solr optimization process rebuilds solr index files, creating a copy of all files. This process improves general Solr performance, but produces high load in server. Solr optimization is necessary when there are a lot of changes in solr index: inserts, updates or deletes. It is not a performance key.

8.2.2. Subscription History

During installation/configuration process, user can activate history system (Read Application Configuration File for more information). When history system is active, the performance decreases because each update or delete operation generates a new insert query to database. Besides, "subscription_hist" table can grow witout limit. Previous point generates two problems: delay of history insert operations (and consequently of update/delete queries) and tablespace growth.

If your SR has a history system active, then Admin should manage old data.

This is a minor point of performance.

8.3. Cluster decision.

If you have completed all previous steps, but performance is low, then you should create a cluster. There are two kinds of cluster: Jboss and Solr. Admin can choose if he desires only a solr cluster or jboss cluster or both.

Jboss cluster it´s necessary when there are a lot of concurrent requests, but persistence system is enough. If performance is poor, but a request number is lineal, then a Solr cluster is a good solution.

8.4. Other sections to improve

During performance test has been detected problems with Oracle. For instance, indexation process times are more expensive with each iteration because each database query is more slower than previous (without user requests and same susbcritpions size). Please, review this point in first place of SR suffers a degradation.

Besides, other performance problems are produced by Solr System (using online indexation) for this reason the times in scheduled mode are really good in comparison with times in online mode. There are several steps to improve performance:

- Use high performance disk for solr and database servers.
- Add more CPU´s or increase speed.
- Use separated server for each purpose: Oracle, Jboss and Solr.
- Use cluster configuration for Jboss and Solr.
- Clusterized Oracle.

Normally, memory consumption is stable and lineal. Limitations are established by CPU and hard disk. If it´s required that server supports a lot of concurrently requests, then both previous factors should be improved. If work load is stable, the best way to get performance (better times) is with high quality disk or SAN (always, it´s necessary hardware monitor to decide the choice or to view hardware limitations).

Verification of signed binary

HP products deliver on our "trustworthy and reliable" brand promise. Creating and delivering an electronic cryptographic "signature" for HP code will give our customers an industry standard method to verify the integrity and authenticity of the code they received from HP before deployment.

The verification of the signed binary will be done by using GnuPG.

For RHEL (Red Hat Enterprise Linux) the GnuPgG is installed by default. You can check check the "gnupg" rpm package is installed with the following command line:

```
rpm -qa "gnupg*"
```

If it is not installed, you can install it from your RHEL media via rpm or yum command.

To verify the signed code, you can use the following command:

```
gpg --verify <.sig file obtained from HPCSS> <input file>*.
```

The output should be as shown similar to one given bellow.

```
gpg: Signature made Wed Nov 17 12:32:46 2010 IST using DSA key ID 2689B887
gpg: Good signature from "Hewlett-Packard Company (HP Codesigning Service)"
Primary key fingerprint: FB41 0E68 CEDF 95D0 6681 1E95 527B C53A 2689 B887
```

NOTE: message "**Good signature from "Hewlett-Packard Company (HP Codesigning Service)**" "indicates the code sign verification is successful.