

Mobicents SIP Presence Service User Guide

by Douglas Silas, Eduardo Martins, Ivelin Ivanov, and Jared Morgan

Preface	v
1. Document Conventions	v
1.1. Typographic Conventions	v
1.2. Pull-quote Conventions	vii
1.3. Notes and Warnings	vii
2. Provide feedback to the authors!	viii
1. Introduction to the Mobicents SIP Presence Service	1
1.1. Architecture of the Mobicents SIP Presence Service	1
2. Installing the Mobicents SIP Presence Service	5
2.1. Mobicents SIP Presence Service: Installing, Configuring and Running	5
2.1.1. Pre-Install Requirements and Prerequisites	5
2.1.2. Downloading	6
2.1.3. Configuring (and Setting JBOSS_HOME)	6
2.1.4. Installing	9
2.1.5. Running	9
2.1.6. Stopping	10
2.1.7. Testing	10
2.1.8. Uninstalling	10
2.1.9. Building from Source Project	11
2.1.10. Binary Releases Daily Snapshots	11
3. Mobicents XML Document Management Server	13
3.1. Functional Architecture of the XDM Server	13
3.2. Configuring the XDM Server	18
3.2.1. Configuring the XDM Server XCAP root	18
3.2.2. XDM Server User Profile Provisioning	19
3.3. Resources and Further Information about the XDM Server	19
4. Mobicents SIP Presence Server	21
4.1. Functional Architecture of the SIP Presence Server	22
4.1.1. Implementation Architecture of the Mobicents SIP Presence Server	24
4.2. Mobicents Resource List Server	26
4.3. Configuring The SIP Presence Server	26
4.4. Resources and Further Information about the SIP Presence Server	26
A. Revision History	27
Index	29

Preface

1. Document Conventions

This manual uses several conventions to highlight certain words and phrases and draw attention to specific pieces of information.

In PDF and paper editions, this manual uses typefaces drawn from the [Liberation Fonts](https://fedorahosted.org/liberation-fonts/) [https://fedorahosted.org/liberation-fonts/] set. The Liberation Fonts set is also used in HTML editions if the set is installed on your system. If not, alternative but equivalent typefaces are displayed. Note: Red Hat Enterprise Linux 5 and later includes the Liberation Fonts set by default.

1.1. Typographic Conventions

Four typographic conventions are used to call attention to specific words and phrases. These conventions, and the circumstances they apply to, are as follows.

Mono-spaced Bold

Used to highlight system input, including shell commands, file names and paths. Also used to highlight key caps and key-combinations. For example:

To see the contents of the file `my_next_bestselling_novel` in your current working directory, enter the `cat my_next_bestselling_novel` command at the shell prompt and press **Enter** to execute the command.

The above includes a file name, a shell command and a key cap, all presented in Mono-spaced Bold and all distinguishable thanks to context.

Key-combinations can be distinguished from key caps by the hyphen connecting each part of a key-combination. For example:

Press **Enter** to execute the command.

Press **Ctrl+Alt+F1** to switch to the first virtual terminal. Press **Ctrl+Alt+F7** to return to your X-Windows session.

The first sentence highlights the particular key cap to press. The second highlights two sets of three key caps, each set pressed simultaneously.

If source code is discussed, class names, methods, functions, variable names and returned values mentioned within a paragraph will be presented as above, in **Mono-spaced Bold**. For example:

File-related classes include `filesystem` for file systems, `file` for files, and `dir` for directories. Each class has its own associated set of permissions.

Proportional Bold

This denotes words or phrases encountered on a system, including application names; dialogue box text; labelled buttons; check-box and radio button labels; menu titles and sub-menu titles. For example:

Choose **System > Preferences > Mouse** from the main menu bar to launch **Mouse Preferences**. In the **Buttons** tab, click the **Left-handed mouse** check box and click **Close** to switch the primary mouse button from the left to the right (making the mouse suitable for use in the left hand).

To insert a special character into a **gedit** file, choose **Applications > Accessories > Character Map** from the main menu bar. Next, choose **Search > Find** from the **Character Map** menu bar, type the name of the character in the **Search** field and click **Next**. The character you sought will be highlighted in the **Character Table**. Double-click this highlighted character to place it in the **Text to copy** field and then click the **Copy** button. Now switch back to your document and choose **Edit > Paste** from the **gedit** menu bar.

The above text includes application names; system-wide menu names and items; application-specific menu names; and buttons and text found within a GUI interface, all presented in Proportional Bold and all distinguishable by context.

Note the **>** shorthand used to indicate traversal through a menu and its sub-menus. This is to avoid the difficult-to-follow 'Select **Mouse** from the **Preferences** sub-menu in the **System** menu of the main menu bar' approach.

Mono-spaced Bold Italic Of Proportional Bold Italic

Whether Mono-spaced Bold or Proportional Bold, the addition of Italics indicates replaceable or variable text. Italics denotes text you do not input literally or displayed text that changes depending on circumstance. For example:

To connect to a remote machine using ssh, type `ssh username@domain.name` at a shell prompt. If the remote machine is `example.com` and your username on that machine is john, type `ssh john@example.com`.

The `mount -o remount file-system` command remounts the named file system. For example, to remount the `/home` file system, the command is `mount -o remount /home`.

To see the version of a currently installed package, use the `rpm -q package` command. It will return a result as follows: `package-version-release`.

Note the words in bold italics above `username`, `domain.name`, `file-system`, `package`, `version` and `release`. Each word is a placeholder, either for text you enter when issuing a command or for text displayed by the system.

Aside from standard usage for presenting the title of a work, italics denotes the first use of a new and important term. For example:

When the Apache HTTP Server accepts requests, it dispatches child processes or threads to handle them. This group of child processes or threads is known as

a *server-pool*. Under Apache HTTP Server 2.0, the responsibility for creating and maintaining these server-pools has been abstracted to a group of modules called *Multi-Processing Modules (MPMs)*. Unlike other modules, only one module from the MPM group can be loaded by the Apache HTTP Server.

1.2. Pull-quote Conventions

Two, commonly multi-line, data types are set off visually from the surrounding text.

Output sent to a terminal is set in `Mono-spaced Roman` and presented thus:

```
books      Desktop  documentation  drafts  mss    photos  stuff  svn
books_tests Desktop1  downloads      images  notes  scripts svgs
```

Source-code listings are also set in `Mono-spaced Roman` but are presented and highlighted as follows:

```
package org.jboss.book.jca.ex1;

import javax.naming.InitialContext;

public class ExClient
{
    public static void main(String args[])
        throws Exception
    {
        InitialContext iniCtx = new InitialContext();
        Object      ref  = iniCtx.lookup("EchoBean");
        EchoHome    home = (EchoHome) ref;
        Echo        echo = home.create();

        System.out.println("Created Echo");

        System.out.println("Echo.echo('Hello') = " + echo.echo("Hello"));
    }
}
```

1.3. Notes and Warnings

Finally, we use three visual styles to draw attention to information that might otherwise be overlooked.



Note

A note is a tip or shortcut or alternative approach to the task at hand. Ignoring a note should have no negative consequences, but you might miss out on a trick that makes your life easier.



Important

Important boxes detail things that are easily missed: configuration changes that only apply to the current session, or services that need restarting before an update will apply. Ignoring Important boxes won't cause data loss but may cause irritation and frustration.



Warning

A Warning should not be ignored. Ignoring warnings will most likely cause data loss.

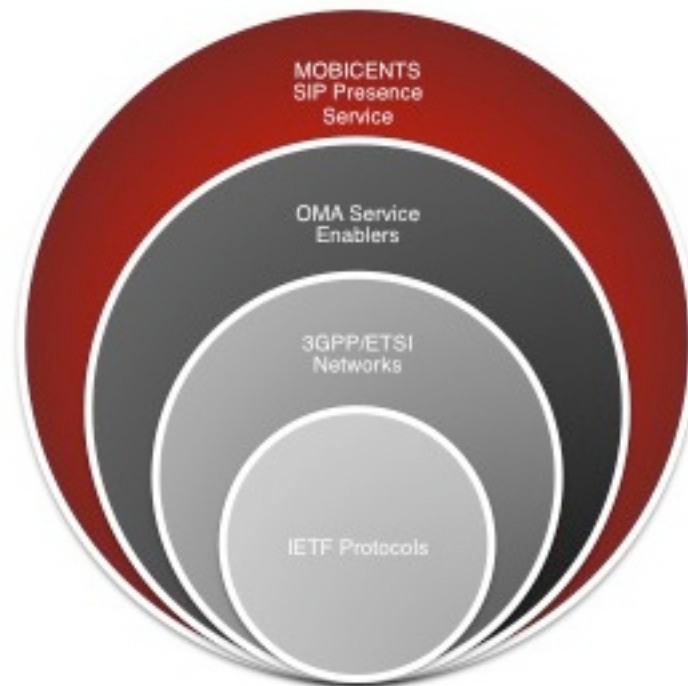
2. Provide feedback to the authors!

If you find a typographical error in this manual, or if you have thought of a way to make this manual better, we would love to hear from you! Please submit a report in Bugzilla: <http://bugzilla.redhat.com/bugzilla/> against the product **SIP Presence Service**, or contact the authors.

When submitting a bug report, be sure to mention the manual's identifier: `SIP_Presence_Service_User_Guide`

If you have a suggestion for improving the documentation, try to be as specific as possible when describing it. If you have found an error, please include the section number and some of the surrounding text so we can find it easily.

Introduction to the Mobicents SIP Presence Service

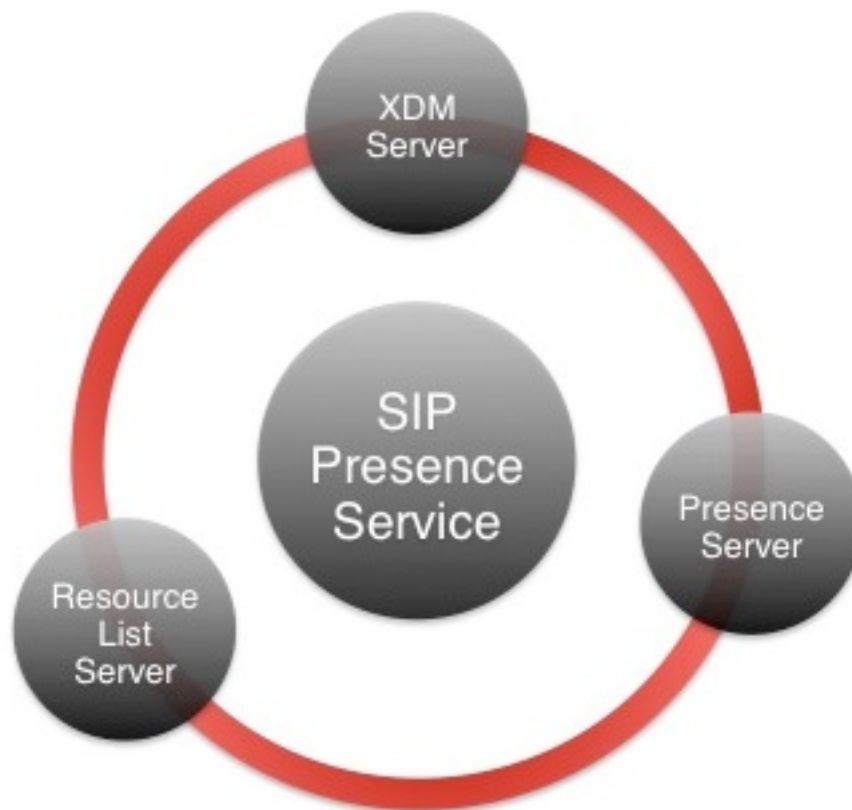


Mobicents SIP Presence Service relation with standard groups

The **Mobicents SIP Presence Service** provides presence functionalities to SIP-based networks using standards developed by the Internet Engineering Task Force (IETF), the Open Mobile Alliance (OMA), the 3rd Generation Partnership Project (3GPP) and the European Telecommunications Standards Institute (ETSI).

1.1. Architecture of the Mobicents SIP Presence Service

The **SIP Presence Service** is comprised of three separate but interrelated servers.



Mobicents SIP Presence Service servers

The Three Servers Comprising the SIP Presence Service

The SIP Presence Server

The **Mobicents SIP Presence Server** (PS) is an entity that accepts, stores and distributes SIP Presence Information. The **Presence Server** performs the following functions:

- It manages publications from one or multiple presence source(s) of a certain *presentity*. This includes refreshing presence information, replacing existing presence information with newly-published information, or removing presence information.
- It manages subscriptions from watchers to presence information and generates notifications about presence information state changes, retrieving the presence authorization rules from the **XDM Server**.
- It manages subscriptions from watcher information subscribers to watcher information and generates notifications about watcher information state changes.

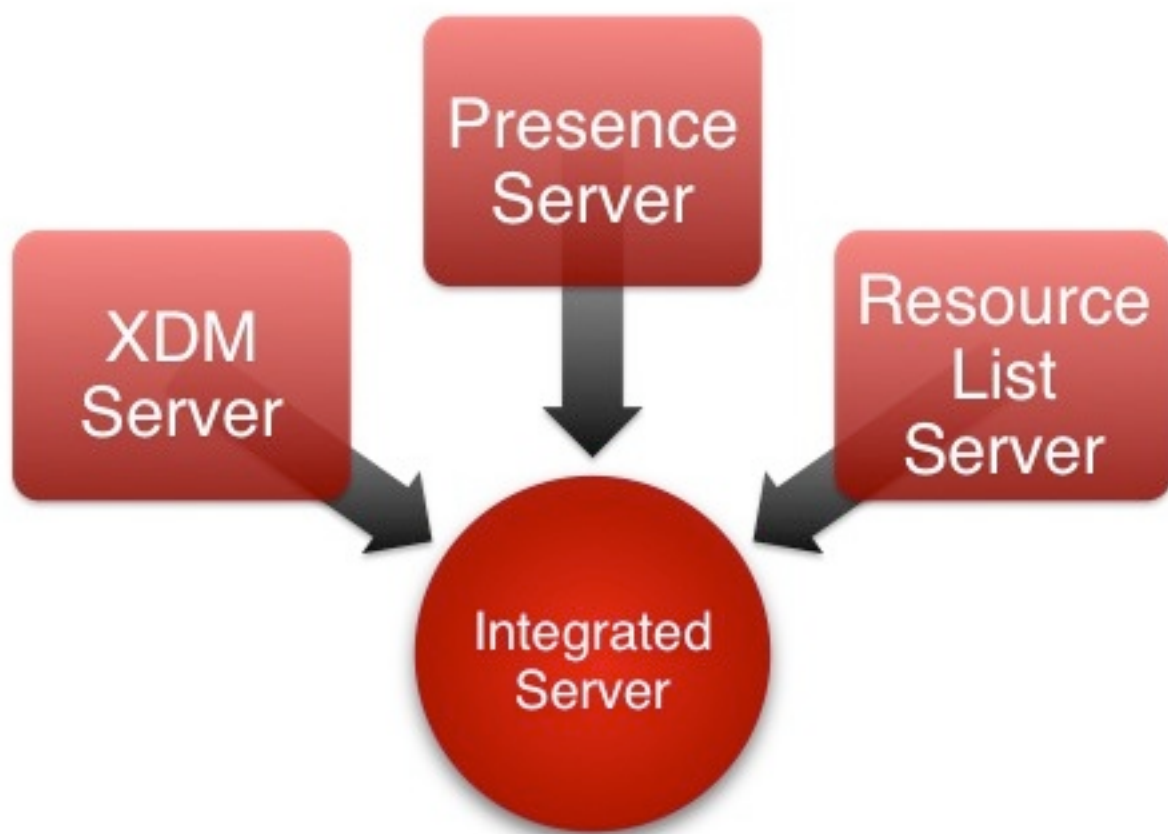
The XML Document Management Server

The **XML Document Management Server** (XDMS) is a functional element of next-generation IP communications networks is responsible for handling the management of user XML

documents stored on the network side, such as presence authorization rules, static presence information, contact and group lists (also known as “resource lists”), policy data, and many others.

The Resource List Server

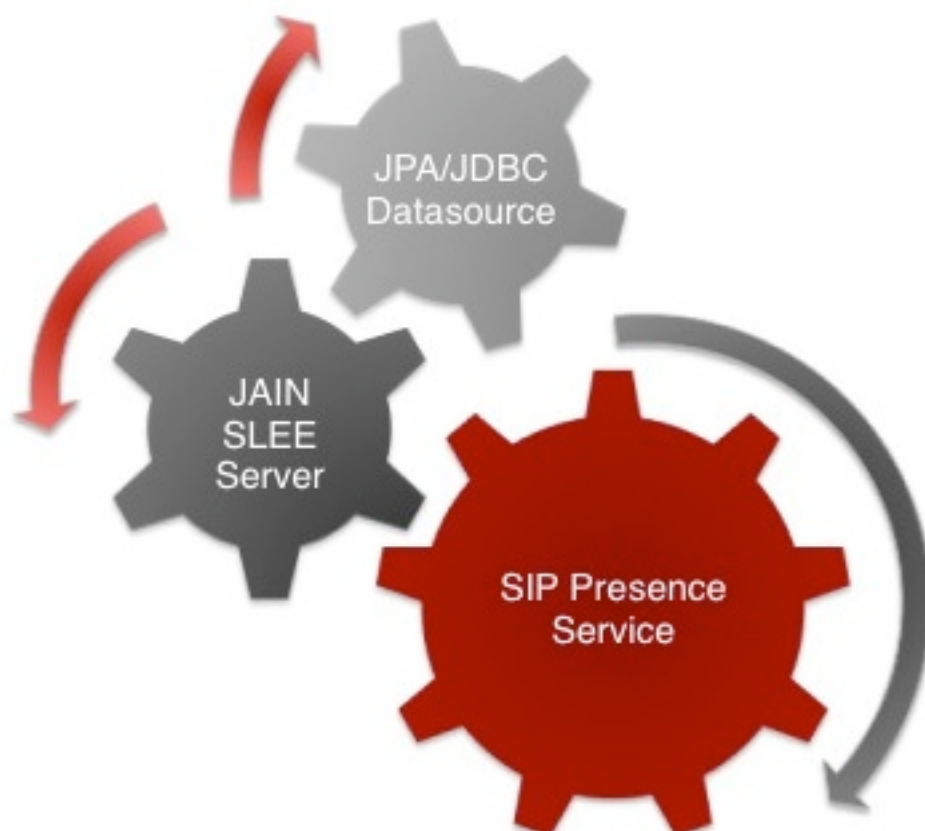
The **Resource List Server** (RLS) handles subscriptions to presence lists. It creates and manages back-end subscriptions to all resources in the presence list. The list content is retrieved from the **XDM Server**.



Mobicents SIP Presence Service Integrated server

A major advantage of the **Mobicents SIP Presence Service** is that, depending on your needs, each server can be deployed separately, or all servers can be integrated on the same host.

The **Mobicents SIP Presence Service** is built on top of **Mobicents JAIN SLEE**, a high performance and scalable Application Server and uses many additional Java Enterprise (JEE) technologies, such as Java Persistence API (JPA) to manage data.



Mobicents SIP Presence Service Integrated implementation

In addition, there are JAIN SLEE internal client interfaces available for interaction with each server, which distinguishes the **Mobicents SIP Presence Service** from other presence services.

Resources and Further Information about the Mobicents SIP Presence Service. For further information on the **Mobicents SIP Presence Service**, here is a list of additional resources:

Sources

[Source Code Location](http://mobicents.googlecode.com/svn/trunk/servers/sip-presence/) [http://mobicents.googlecode.com/svn/trunk/servers/sip-presence/]

Community

[Mobicents Community](http://groups.google.com/group/mobicents-public) [http://groups.google.com/group/mobicents-public]

Installing the Mobicents SIP Presence Service

2.1. Mobicents SIP Presence Service: Installing, Configuring and Running

There are multiple binary distributions of the **Mobicents SIP Presence Service**.

Description of the different Mobicents SIP Presence Service Distributions

The **Integrated SIP Presence Service** binary distribution *with Mobicents JAIN SLEE*

These installation instructions detail the installation, running and configuring of the *Integrated* binary **Mobicents SIP Presence Service** distribution. This distribution includes the **XDM** and **SIP Presence Servers**, the servers are pre-installed in a version of the **Mobicents JAIN SLEE Server**, and the Mobicents JAIN SLEE SIP11 and HTTP Servlet Resource Adaptors. Examples of JAIN SLEE applications interacting with the **Mobicents Integrated SIP Presence Service** are also included and come pre-installed.

The stand-alone **Mobicents XDM Server** binary distribution *with Mobicents JAIN SLEE*

Users who wish to deploy the **Mobicents XML Document Server** on a different host or who do not require the **Mobicents Presence Server** should install the stand-alone **Mobicents XDM Server** binary distribution. The following installation, running and configuring instructions provide parallel instructions specific to the **Mobicents XDM Server**.

The **Mobicents SIP Presence Service** binary distribution *without Mobicents JAIN SLEE*

Users who have already installed and set up a separate **Mobicents JAIN SLEE** installation may want to install one or more servers of the **Mobicents SIP Presence Service**.

2.1.1. Pre-Install Requirements and Prerequisites

You should ensure that a few requirements have been met before continuing with the install.

Hardware Requirements

Anything Java Itself Will Run On

The **Mobicents SIP Presence Service** is an 100% Java application. **Mobicents SIP Presence Service** will run on the same hardware that the **Mobicents JAIN SLEE** runs on.

Software Prerequisites

JDK 5

A working installation of the Java Development Kit (JDK) version 5 is required in order to run the **Mobicents SIP Presence Service**.



Note

JDK 6 is not supported by the **Mobicents XDM Server**

Apache Ant 1.6 or later

A working installation of the Apache Ant 1.6 or later is required in order to install the **Mobicents SIP Presence Service** release without **\$PLATFORM_NAME JAIN SLEE**.

JBOSS_HOME Environment Variable

The environment variable JBOSS_HOME, if set, must be pointing to the **JBoss AS** within **Mobicents JAIN SLEE**.

2.1.2. Downloading

You can download the latest version of the **Mobicents SIP Presence Service** distribution you need from the Mobicents Downloads page at http://sourceforge.net/project/showfiles.php?group_id=102670. Click on the Mobicents SIP Presence Service link to view all available distributions and downloads. The latest releases are nearer the top.

If you are unsure which distribution zip file to download, refer to [Description of the different Mobicents SIP Presence Service Distributions](#), and then to the following list of release binaries.

Mobicents SIP Presence Service Binary Distribution Zip Files

mobicents-sip-presence-integrated-1.0.0.BETA5.zip

Download this zip file to obtain the **Mobicents Integrated SIP Presence Service** binary distribution, which includes the **Mobicents SIP Presence Server**, the **Mobicents XDM Server**, and the **JBoss Application Server with Mobicents JAIN SLEE**, well as all required JAIN SLEE Resource Adaptors.

mobicents-sip-presence-xdms-1.0.0.BETA5.zip

Download this zip file to obtain the **Mobicents XDM Server** binary distribution, which bundles the **JBoss Application Server with Mobicents JAIN SLEE**.

2.1.3. Configuring (and Setting JBOSS_HOME)

2.1.3.1. Setting the JBOSS_HOME Environment Variable

The **Mobicents Platform (Mobicents)** is built on top of the **JBoss Application Server (JBoss AS)**. You do not need to set the JBOSS_HOME environment variable to run any of the **Mobicents Platform** servers *unless* JBOSS_HOME is *already* set.

The best way to know for sure whether JBOSS_HOME was set previously or not is to perform a simple check which may save you time and frustration.

Checking to See If JBOSS_HOME is Set on Unix. At the command line, `echo $JBOSS_HOME` to see if it is currently defined in your environment:

```
~]$ echo $JBOSS_HOME
```

The **Mobicents Platform** and most Mobicents servers are built on top of the **JBoss Application Server (JBoss AS)**. When the **Mobicents Platform** or Mobicents servers are built *from source*, then `JBOSS_HOME` *must* be set, because the Mobicents files are installed into (or “over top of” if you prefer) a clean **JBoss AS** installation, and the build process assumes that the location pointed to by the `JBOSS_HOME` environment variable at the time of building is the **JBoss AS** installation into which you want it to install the Mobicents files.

This guide does not detail building the **Mobicents Platform** or any Mobicents servers from source. It is nevertheless useful to understand the role played by **JBoss AS** and `JBOSS_HOME` in the Mobicents ecosystem.

The immediately-following section considers whether you need to set `JBOSS_HOME` at all and, if so, when. The subsequent sections detail how to set `JBOSS_HOME` on Unix and Windows



Important

Even if you fall into the category below of *not needing* to set `JBOSS_HOME`, you may want to for various reasons anyway. Also, even if you are instructed that you do *not need* to set `JBOSS_HOME`, it is good practice nonetheless to check and make sure that `JBOSS_HOME` actually *isn't* set or defined on your system for some reason. This can save you both time and frustration.

You **DO NOT NEED** to set `JBOSS_HOME` if...

- ...you have installed the **Mobicents Platform** binary distribution.
- ...you have installed a Mobicents server binary distribution *which bundles JBoss AS*.

You **MUST** set `JBOSS_HOME` if...

- ...you are installing the **Mobicents Platform** or any of the Mobicents servers *from source*.
- ...you are installing the **Mobicents Platform** binary distribution, or one of the Mobicents server binary distributions, which *do not* bundle **JBoss AS**.

Naturally, if you installed the **Mobicents Platform** or one of the Mobicents server binary releases which *do not* bundle **JBoss AS**, yet requires it to run, then you should [install JBoss AS](http://www.jboss.org/file-access/default/members/jbossas/freezone/docs/Installation_Guide/4/html/index.html) [http://www.jboss.org/file-access/default/members/jbossas/freezone/docs/Installation_Guide/4/html/index.html] before setting `JBOSS_HOME` or proceeding with anything else.

Setting the JBOSS_HOME Environment Variable on Unix. The `JBOSS_HOME` environment variable must point to the directory which contains all of the files for the **Mobicents Platform** or individual Mobicents server that you installed. As another hint, this topmost directory contains a `bin` subdirectory.

Setting `JBOSS_HOME` in your personal `~/.bashrc` startup script carries the advantage of retaining effect over reboots. Each time you log in, the environment variable is sure to be set for you, as a user. On Unix, it is possible to set `JBOSS_HOME` as a system-wide environment variable, by defining it in `/etc/bashrc`, but this method is neither recommended nor detailed in these instructions.

Procedure 2.1. To Set JBOSS_HOME on Unix...

1. Open the `~/.bashrc` startup script, which is a hidden file in your home directory, in a text editor, and insert the following line on its own line while substituting for the actual install location on your system:

```
export JBOSS_HOME="/home/<username>/<path>/<to>/<install_directory>"
```

2. Save and close the `.bashrc` startup script.
3. You should `source` the `.bashrc` script to force your change to take effect, so that `JBOSS_HOME` becomes set for the current session¹.

```
~]$ source ~/.bashrc
```

4. Finally, ensure that `JBOSS_HOME` is set in the current session, and actually points to the correct location:



Note

The command line usage below is based upon a binary installation of the **Mobicents Platform**. In this sample output, `JBOSS_HOME` has been set correctly to the `topmost_directory` of the **Mobicents** installation. Note that if you are installing one of the standalone **Mobicents** servers (with **JBoss AS** bundled!), then `JBOSS_HOME` would point to the `topmost_directory` of your server installation.

```
~]$ echo $JBOSS_HOME
/home/silas/
```

Setting the JBOSS_HOME Environment Variable on Windows. The `JBOSS_HOME` environment variable must point to the directory which contains all of the files for the Mobicents

¹ Note that any other terminals which were opened prior to your having altered `.bashrc` will need to `source` `~/.bashrc` as well should they require access to `JBOSS_HOME`.

Platform or individual Mobicents server that you installed. As another hint, this topmost directory contains a `bin` subdirectory.

For information on how to set environment variables in recent versions of Windows, refer to <http://support.microsoft.com/kb/931715>.

2.1.4. Installing

Once the requirements and prerequisites have been met, and you have downloaded the correct zip file for the binary distribution you need, you are ready to install, please follow the instructions below.

Procedure 2.2. Installing a Mobicents SIP Presence Service Binary Distribution *with* Mobicents JAIN SLEE

1. Unzip the release file
2. Ensure that the environment variable `JBOSS_HOME` is either not set, or pointing to the `<install_directory>` directory extracted from the release file.

Procedure 2.3. Installing a Mobicents SIP Presence Service Binary Distribution *without* Mobicents JAIN SLEE

1. Unzip the release file
2. Ensure that the environment variable `JBOSS_HOME` is set and pointing to the JBoss AS with the **Mobicents JAIN SLEE**, where **Mobicents SIP Presence Service** components should be installed.
3. Invoke the correct Apache Ant target in `build.xml` script to install:

Install the Integrated SIP Presence Service:

```
~]$ ant integrated-deploy
```

Install the stand-alone XDM Server:

```
~]$ ant xdms-deploy
```

2.1.5. Running

Once installed, you can run server(s) by executing the `run.sh` (Unix) or `run.bat` (Microsoft Windows) startup scripts in the `<install_directory>/bin` directory (on Unix or Windows). In the Unix

terminal or Command Prompt, you will be able to tell that the server started successfully if the last line of output is similar to the following (ending with “Started in 23s:648ms”):

```
11:23:07,656 INFO [Server] JBoss (MX MicroKernel) [4.2.3.GA (build:
SVNTag=JBoss_4_2_2_GA date=200710221139)] Started in 23s:648ms
```



Note

By default the server(s) start and bind to 127.0.0.1 IP, to use a different hostname or IP use the -b HOST parameter when executing the startup script (e.g. run.sh -b 172.31.1.1).

2.1.6. Stopping

You can shut down the server(s) you can run server(s) by executing the shutdown.sh (Unix) or shutdown.bat (Microsoft Windows) scripts in the <install_directory>/bin directory (on Unix or Windows). Note that if you properly stop the server, you will see the following three lines as the last output in the Unix terminal or Command Prompt:

```
[Server] Shutdown complete
Shutdown complete
Halting VM
```

2.1.7. Testing

TODO

2.1.8. Uninstalling

Procedure 2.4. Uninstalling a Mobicents SIP Presence Service Binary Distribution bundled *with* Mobicents JAIN SLEE

- To uninstall the **SIP Presence Service** or **XDM Server**, simply delete the directory you decompressed the binary distribution archive into.

Procedure 2.5. Uninstalling a Mobicents SIP Presence Service Binary Distribution *without* Mobicents JAIN SLEE

- Invoke the correct Apache Ant target in build.xml script to uninstall:

Uninstall the Integrated SIP Presence Service:

```
~]$ ant integrated-undeploy
```

Uninstall the stand-alone XDM Server:

```
~]$ ant xdms-undeploy
```

2.1.9. Building from Source Project



Note

The source building process requires access to the Internet. It also requires an **SVN Client** (to sources checkout) and **Apache Maven2 2.0.9+** (for the building process) installed.

The source project can be downloaded using SVN, the checkout URL is <http://mobicents.googlecode.com/svn/tags/servers/sip-presence/mobicents-sip-presence-service-1.0.0.BETA5>

To build the binaries from source, enter the release directory inside the directory used to checkout the source project and:

```
~]$ ant
```

2.1.10. Binary Releases Daily Snapshots

Everyday a binary release snapshot is built using current sources in development trunk, those are accessible from <http://hudson.jboss.org/hudson/view/Mobicents/job/MobicentsSipPresenceRelease/>

Mobicents XML Document Management Server

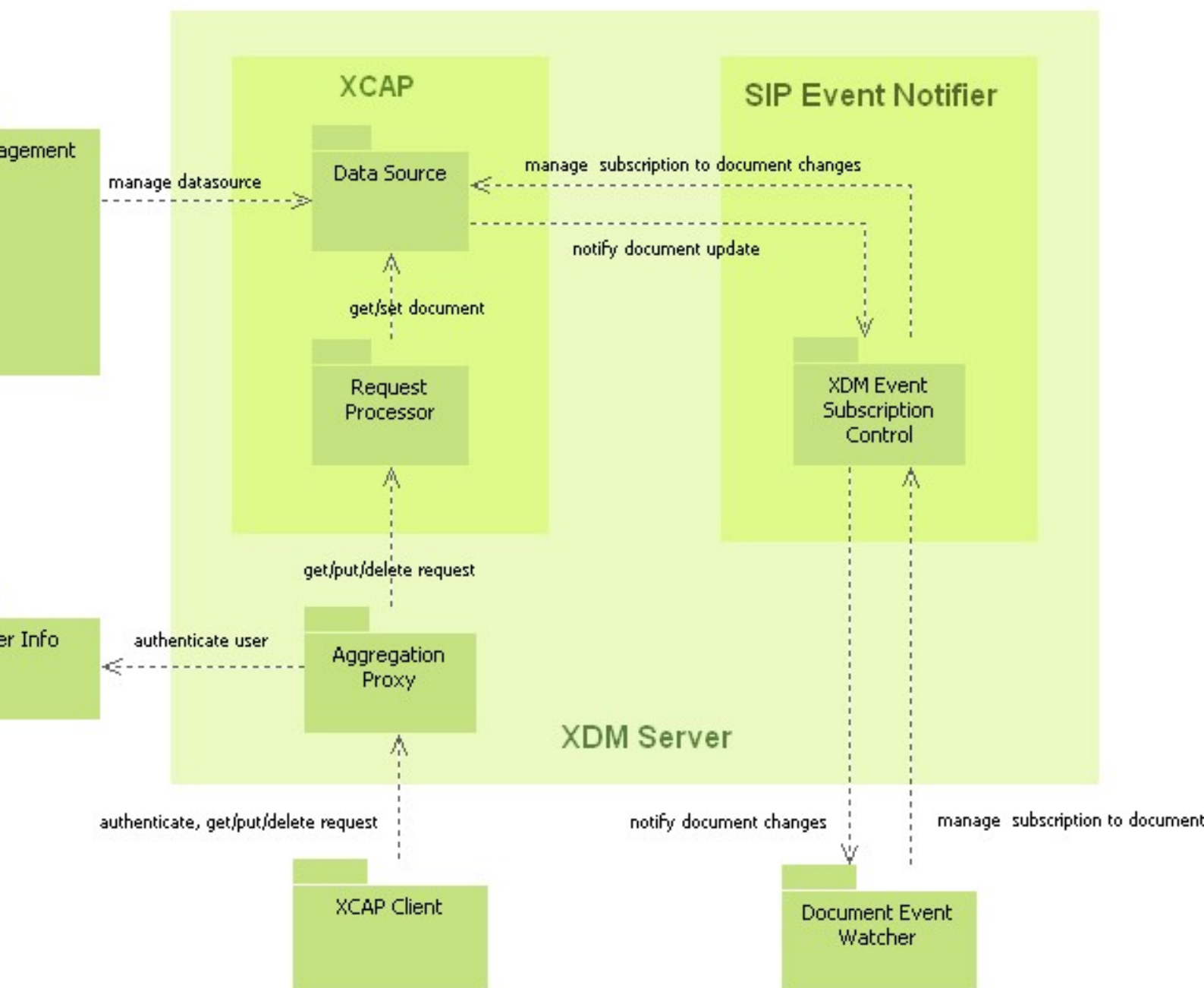
The **Mobicents XML Document Management Server (XDM Server)** is part of the **Mobicents SIP Presence Service**; it is the first free and open source implementation of an **XML Document Management Server** as defined in the [Open Mobile Alliance \(OMA\) XML Document Management v1.1 specification](#). This functional element of next-generation IP communication networks is responsible for handling the management of user XML documents stored on the network side, such as presence authorization rules, contact and group lists (also known as resource lists), static presence information, and much more.

3.1. Functional Architecture of the XDM Server

The **Mobicents XDM Server** includes the following XCAP application usages:

- [IETF Presence Rules \(RFC 5025\)](http://tools.ietf.org/html/rfc5025) [<http://tools.ietf.org/html/rfc5025>]
- [OMA Presence Rules \(OMA Presence Simple v1.1 Candidate Release\)](http://www.openmobilealliance.org/Technical/release_program/Presence_simple_v1_1.aspx) [http://www.openmobilealliance.org/Technical/release_program/Presence_simple_v1_1.aspx]
- [IETF Resource Lists](http://tools.ietf.org/html/rfc4826) [<http://tools.ietf.org/html/rfc4826>]
- [IETF RLS Services \(RFC 4826\)](http://tools.ietf.org/html/rfc4826) [<http://tools.ietf.org/html/rfc4826>]
- [IETF XCAP-CAPS \(RFC 4825\)](http://tools.ietf.org/html/rfc4825) [<http://tools.ietf.org/html/rfc4825>]

The SIP interface partially implements the XCAP Diff Event IETF draft, version 3. Subscriptions to a single document or usage by an entire application are supported. However, these differing usages do not extend to the single-XML element or attribute value level. Regarding the notifications, the diff-processing subscription parameter, if present, is ignored, and patching of content is not available at the moment, which means that only the document etags, new and/or old, will be provided.



The Mobicents XML Document Management Server

The **XDM Server** comprises the following functional elements:

Functional Elements of the XDM Server

Data Source

The **XDM Server** data source is where all user XML documents are stored. Information related to the server itself is also stored in this element along with the user's provisioned data

The data source also handles subscriptions to updates on specific documents, or complete XCAP application usages.

Aggregation Proxy

The aggregation proxy is responsible for handling an XDM client's XCAP requests, which includes authentication of the requester.

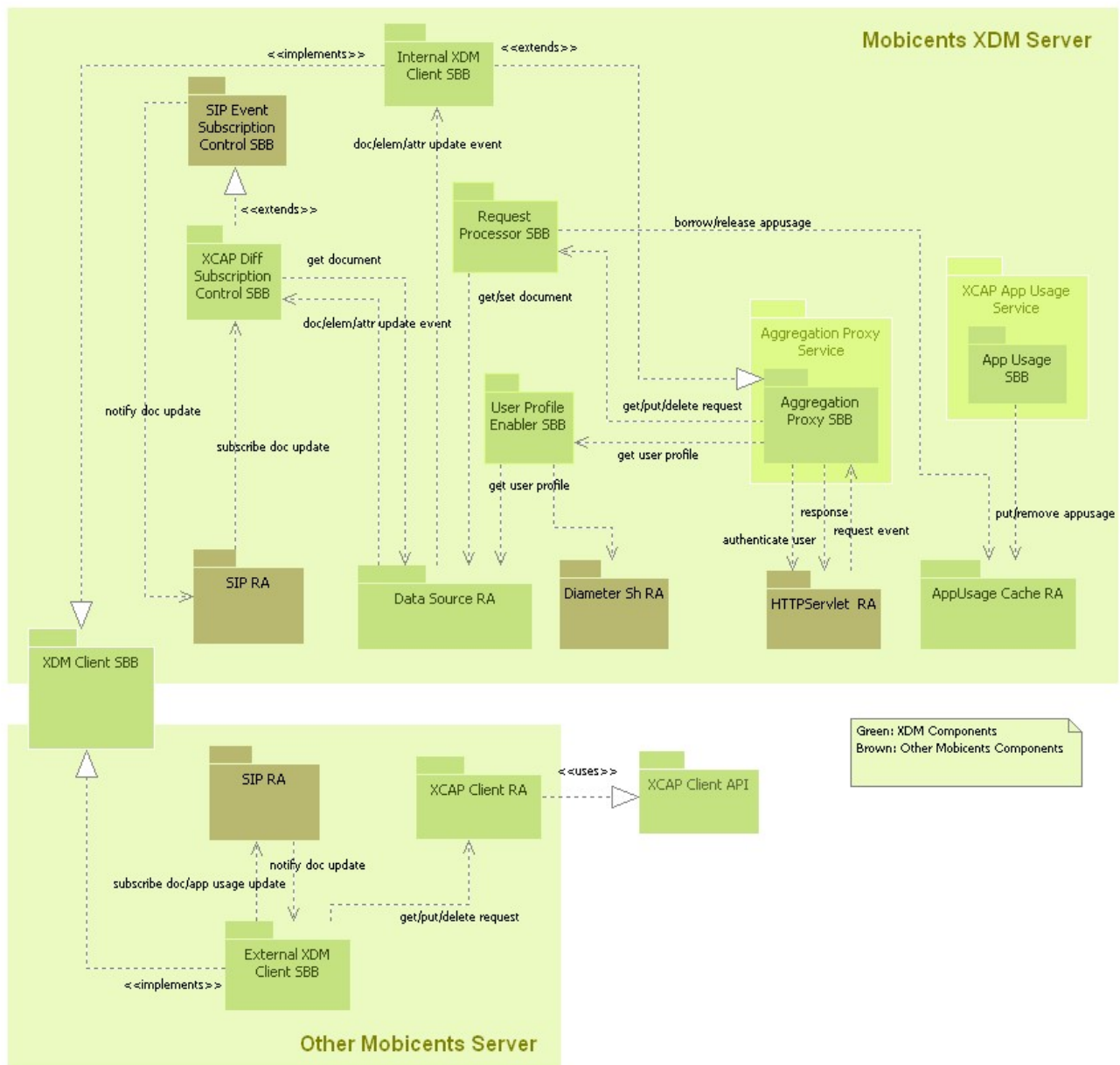
Request Processor

This element includes the XCAP Server logic to process an XCAP request and return a proper response, including authorization for the authenticated user.

XDM Event Subscription Control

This element, using the SIP protocol, is responsible for handling subscriptions to documents managed by the XDM. Its functions include the authentication and authorization of a subscription, attachment to update events on specific documents or application usages, and the sending of notifications when documents change.

Implementation Architecture of the Mobicents XML Document Management Server. The **XDM Server** is built on top of the **Mobicents JAIN SLEE** container. This figure depicts the architecture of the **XDM Server** implementation.



Mobicents XML Document Management Server Implementation

The Functional Elements of the XML Document Management Server

Data Source Resource Adapter

This resource adapter implements the Data Source functional element.

The `RA Type` defines two activities objects, `DocumentActivity` and `AppUsageActivity`, both of which are used to fire events that signal that a document, element or attribute was updated.

The `RA Type` also defines a Service Building Block (SBB) RA interface to manage the users and documents stored in the XDM Server, and create activities, where events will be fired.

The resource adapter will only fire events on activities that exist; that is, the RA won't create activities implicitly if a document is updated.

The `RA_Type` also provides a base abstract implementation of the resource adapter, making it very simple to change the underlying resource used to store information, which is by default the internal JDBC datasource of the **JBoss Application Server**.

AppUsage Cache Resource Adaptor

This resource adapter stores the XCAP application usages installed in the server.

Each `AppUsage` is an object that includes the logic to validate XCAP documents that result from XCAP requests and are expensive to create; this resource adapter thus provides caching of AppUsages, using a pool model.

The resource adapter doesn't possess events or activities.

AppUsage Service

XCAP Application Usages are installed through a JAIN SLEE service, making it possible to add and/or remove application usages while the server is running.

Aggregation Proxy Service

This JAIN SLEE service implements the aggregation proxy functional element. It handles events fired by the Mobicents HTTP Servlet resource adapter and then uses two child SBBs: the `User Profile Enabler SBB` to retrieve information regarding the user needed for authentication/authorization of the XCAP request, and the `Request Processor SBB`, which handles the XCAP request.

Request Processor SBB

The `Request Processor SBB` implements the request processor functional element, providing a synchronous SBB interface to process XCAP requests. It uses the `AppUsage Cache` resource adapter to borrow `AppUsage` objects, and the `Data Source` resource adapter to retrieve or set documents stored in the server's data source.

User Profile Enabler SBB

This SBB provides a synchronous SBB interface used in JAIN SLEE child relations in order to retrieve user information, to be used on user authentication. Two different implementations of the interface are provided: the first considers whether the information is stored in the XDM Data Source, another interfaces with a Diameter Sh Server, such as **IMS HSS**.

XCAP Diff Subscription Control Service

This JAIN SLEE Service extends the abstract SIP Event Subscription Control component to handle SIP subscriptions on the xcap-diff event package.

The implementation architecture figure also contains client-side components:

Client-Side Components of the XML Document Management Server

XCAP Client

The XCAP client is a simple API to interact with an XCAP Server that internally uses the Apache HTTP Client.

XCAP Client Resource Adaptor

The XCAP Client Resource Adapter adapts the XCAP Client API into the JAIN SLEE domain. It provides methods to interact with the XCAP server in both synchronous and asynchronous ways.

The RA Type description and code snippets using the RA can be found [here](#).

XDM Client SBB

The XDMClientSBB is an interface of a JAIN SLEE SBB to be used as a client to the **Mobicents XDM Server** (and others compliant with same standards), in JAIN SLEE child relations.

Two implementations of this interface are provided:

- `InternalXDMClientSBB` is intended to be used on applications running in the **Mobicents XDM Server** JAIN SLEE container, and
- `ExternalXDMClientSBB`, which is intended to be used on applications in a different JAIN SLEE container than the **Mobicents XDM Server**.

3.2. Configuring the XDM Server

3.2.1. Configuring the XDM Server XCAP root

The Mobicents XDM Server comes pre-configured for an XCAP root of `http://<hostname>:8080/mobicents`, hostname being the host/IP used to start the server (127.0.0.1 by default). It is possible to change the last path segment:

- Rename `$JBASS_HOME/server/<server_profile>/deploy/mobicents.war` to the desired last path segment in the XCAP root (e.g. rename to `xcap-root.war` for an XCAP root of `http://<hostname>:8080/xcap-root`). The `<server_profile>` is the server configuration/profile used in the underlying **JBoss AS**, by default it is *default*
- Edit properties file `$JBASS_HOME/server/<server_profile>/deploy/msps-xdms-ds-ra-DU-1.0.0.BETA5.jar/library/msps-xdms-core-sbb-1.0.0.BETA5.jar/org/mobicents/slee/xdm/server/configuration.properties`, define the new XCAP root last path segment through the property `XCAP_ROOT`, don't forget to use a leading `/`. Again the `<server_profile>` is the server configuration/profile used in the underlying **JBoss AS**.

3.2.2. XDM Server User Profile Provisioning

XCAP interface is public, used by users to manage their information such as buddy list, presence authorization rules, etc. thus it needs to enforce user authentication. To do the user authentication, the server relies on the User Profile Enabler managed data, such as the users passwords, and this information must be provisioned, this can be done in two ways, both requiring the server to be running:

User Provisioning through an JMX Client

Users can be added/removed through the MBean named *slee:userprofile=UserProfileControl*

User Provisioning through the **JBoss AS** default datasource.

Users can be added/removed through adding/removing rows of the table named *MOBICENTS_SLEE_ENABLER_USERPROFILES*.

3.3. Resources and Further Information about the XDM Server

For further information on the Mobicents XDM Server, here is a list of additional resources:

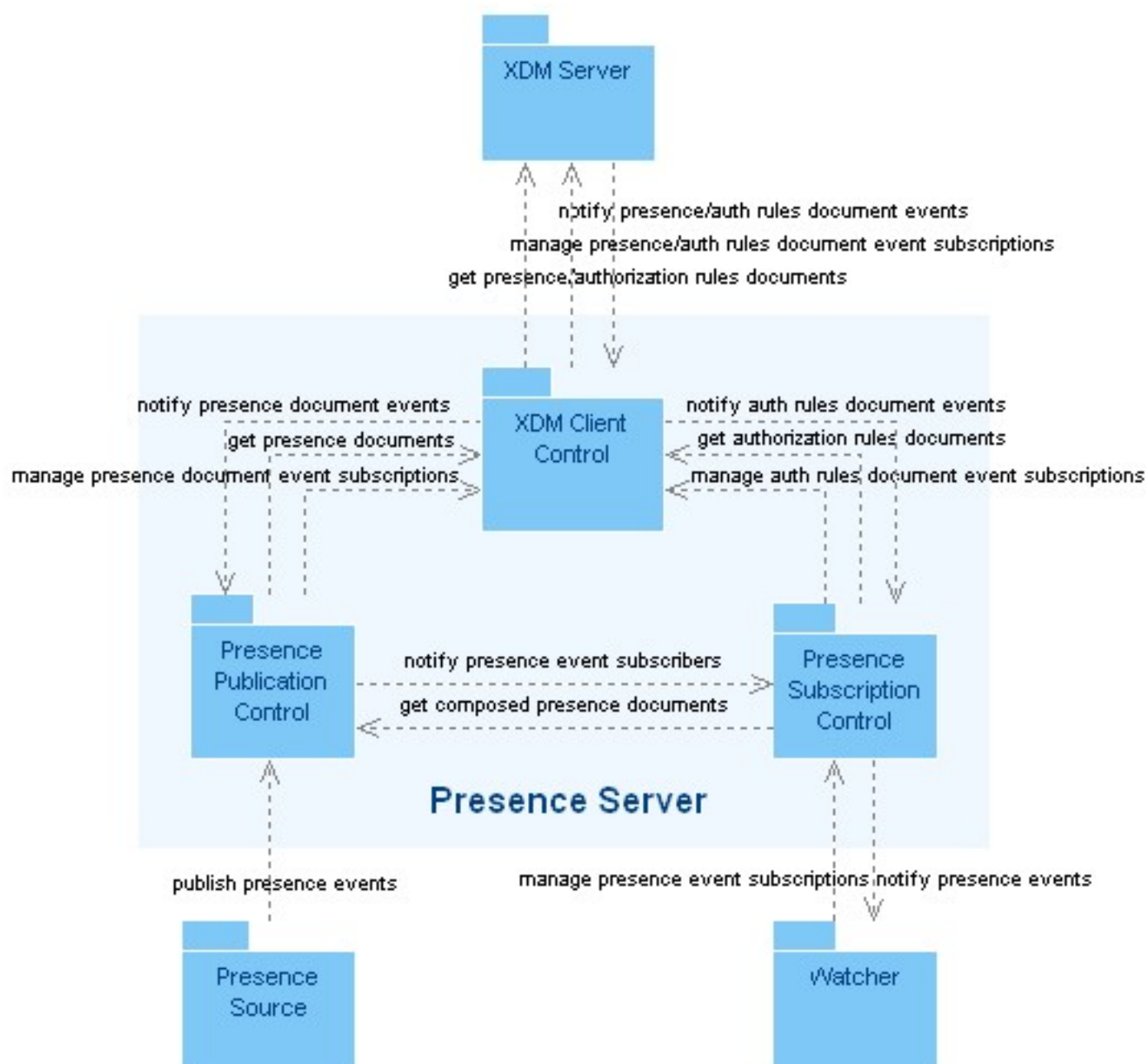
- [XCAP Client API Documentation](http://groups.google.com/group/mobicents-public/web/xcap-client-api) [http://groups.google.com/group/mobicents-public/web/xcap-client-api]
- [XCAP Client RA Type Description and Example Code Snippets](http://groups.google.com/group/mobicents-public/web/xcap-client-resource-adaptor) [http://groups.google.com/group/mobicents-public/web/xcap-client-resource-adaptor]

Mobicents SIP Presence Server

The **Mobicents SIP Presence Server** is a free and open source implementation of a SIP Presence Server, as defined by the Internet Engineering Task Force (IETF), the Open Mobile Alliance (OMA), the 3rd Generation Partnership Project (3GPP) and the European Telecommunications Standards Institute (ETSI).

The **SIP Presence Server** is an entity that accepts, stores and distributes SIP presence information.

4.1. Functional Architecture of the SIP Presence Server



Functional Diagram of the Mobicents SIP Presence Server

The **SIP Presence Server** is comprised of the following functional elements:

Presence Publication Control

This functional element manages the publication of presence events, which includes not only the handling of new publications, but also the refreshing, modification or removal of, already-published information.

Because the presence resource, which is also called a *presentity*, can have multiple publications simultaneously, such as some state published by a user agent or device, and some location data published by a Presence Network Agent (on behalf of the presentity), this element is also responsible for composing all of the different publications for the same resource.

In some presence networks, it may be of interest to allow resources to have a static presence state which is stored in the XDM Server. In cases like these, Presence Publication Control may need to interface with the **XDM Server** to retrieve and subscribe to (learn about changes to) that information, and use it when composing the final presence information document.

Presence Subscription Control

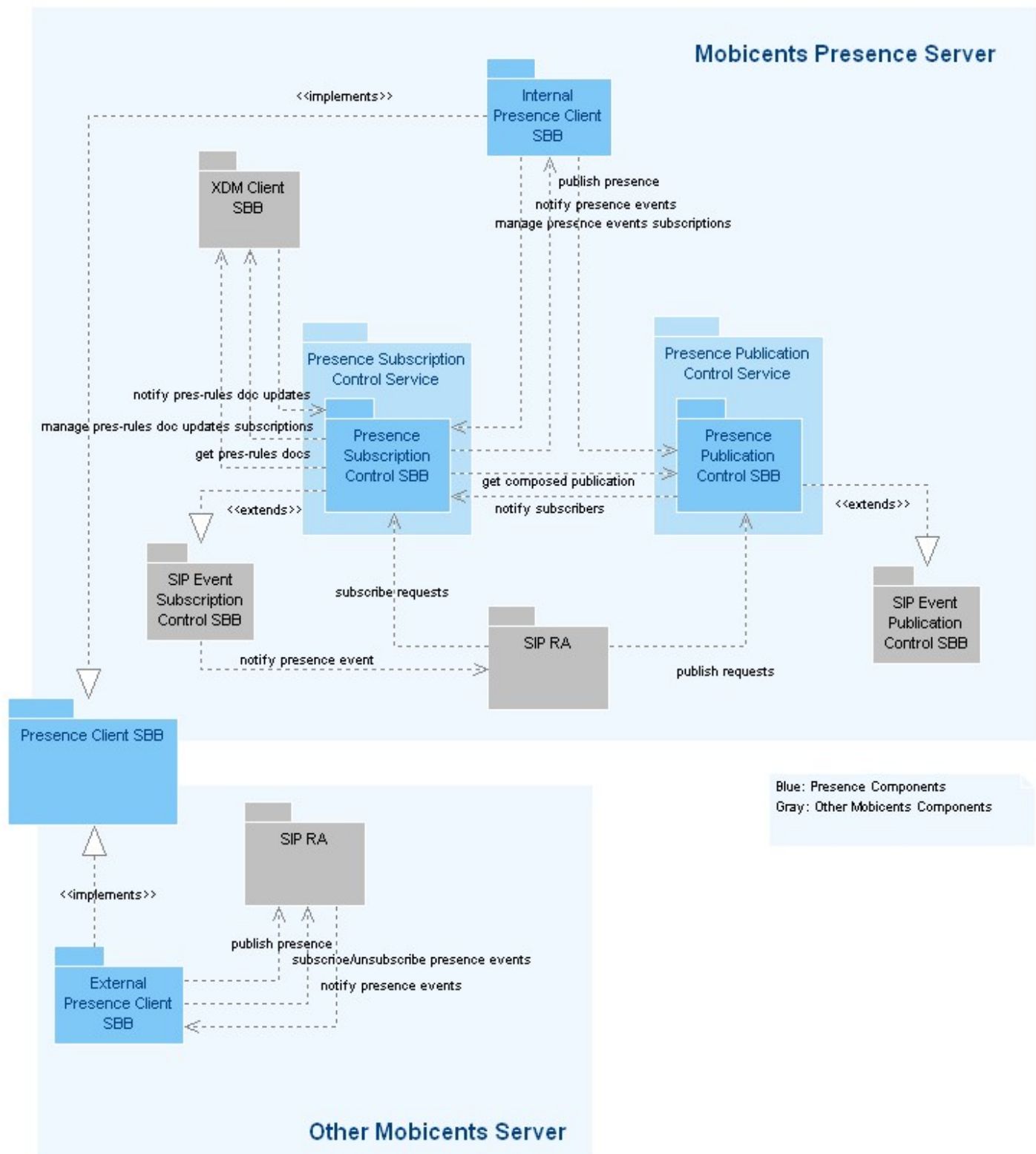
This functional element handles subscriptions to presence events or to the list of subscribers (watchers), for any specific resource. It is, of course, responsible for emitting notifications related to those subscriptions.

Presence authorization rules, which define if a subscription is allowed or rejected and, if allowed, define which transformations to the original presence events are needed, are stored on the **XDM Server** by the user. Thus, Presence Subscription Control needs to retrieve and subscribe to that information.

XDM Client Control

This last element is responsible for interfacing with the **XDM Server** that manages the user's XML documents, and is related to the main functions of the presence server. It is capable not only of retrieving a document or part of one, but also of subscribing to either updates of a single, specific document, or to a full collection of documents of a specific type or application.

4.1.1. Implementation Architecture of the Mobicents SIP Presence Server



Implementation Architecture of the Mobicents SIP Presence Server

The implementation of the Mobicents SIP Presence Server comprises the following functional elements:

The Two Services Which Compose the SIP Presence Server

Presence Publication Control Service

This JAIN SLEE service includes the root Service Building Block (SBB), `PresencePublicationControlSbb`, which is the implementation of the abstract SIP event `PublicationControlSbb`. It handles publications on the **presence** event package.

The `PresencePublicationControlSbb` provides the following capabilities:

- It provides the logic to authorize a publication; however, it only authorizes `PUBLISH` requests when the request URI matches the PIDF document “entity” attribute.
- It provides JAXB unmarshallers to validate and parse the PIDF document for the abstract `PublicationControlSbb`.
- It demands that notifying subscribers occur through a child relation to the root SBB of the Presence Subscription Control Service.
- Finally, it also provides an `SbbLocalObject` interface that can be used, in JAIN SLEE child relations, to obtain the composed presence information for a specific resource.

Presence Subscription Control Service.

This JAIN SLEE service includes the root SBB `PresenceSubscriptionControlSbb`, which is the implementation of the abstract SIP Event `SubscriptionControlSbb`. It handles subscriptions on the “presence” event package.

The standout SBB logic item is the usage of presence-rules documents, obtained through the XDM Client SBB child relation, in order to authorize subscriptions and transform the content notified¹. It also defines a child relation to the root SBB of `PresencePublicationService` to retrieve the composed PIDF document for the subscription's notifier.

The SBB also provides an `SbbLocalObject` interface that can be used, in JAIN SLEE child relations, to make the presence event known to the subscribers of a specific resource.

The implementation architecture of the SIP Presence Server also contains client-side components:

Presence Client SBB

The `PresenceClientSBB` is the interface to a JAIN SLEE SBB intended to be used as a client for the **Mobicents SIP Presence Server** (and other servers compliant with same standards), in JAIN SLEE child relations.

¹Note that this feature is not yet used.

Two implementations of this interface are provided: the `InternalPresenceClientSBB` that is used with applications running in the **Mobicents SIP Presence Server** JAIN SLEE container, and the `ExternalPresenceClientSBB`, used with applications running in a different JAIN SLEE container than the **Mobicents SIP Presence Server**.

4.2. Mobicents Resource List Server

The **Mobicents Resource List Server** is currently tightly integrated with the **Mobicents SIP Presence Server**, supporting RFC 5367 and OMA service uri template.

4.3. Configuring The SIP Presence Server

TODO how to configure the SIP Publication and Subscription Interfaces

4.4. Resources and Further Information about the SIP Presence Server

For further information on the Mobicents SIP Presence Server, see the following list of additional resources:

- [Mobicents SIP Event Publication and Subscription Control Components](http://groups.google.com/group/mobicents-public/web/mobicents-sip-event-components) [<http://groups.google.com/group/mobicents-public/web/mobicents-sip-event-components>]

Appendix A. Revision History

Revision History

Revision 3.0	Fri Jul 10 2009	EduardoMartins
Major update to include XDM Server authentication and authorization, creation of XCAP Application Usages, SIP Client configuration examples and Resource List Server.		
Revision 2.0	Fri Mar 06 2009	DouglasSilas
First release of the "parameterized" and much-improved Mobicents documentation.		
Revision 1.0	Tue Jan 20 2009	DouglasSilas
Creation of the Mobicents SIP Presence User Guide separate from the Mobicents Platform User Guide.		

Index

A

architecture, 1

F

feedback, viii

