# Administrator Guide

**JWebSocket**

**Continuous Integration Environment for jWebSocket**

**V 1.0**

**Version History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Autor** |
| 5/06/2011 | 1.0 | Creation of document | Carlos Alberto Feyt Salgueiro |
| 12/06/2012 | 1.1 | Fix Alexander Feedback | Carlos Alberto Feyt Salgueiro |

**Introduction**

Continuous Integration, also known as CI, is a cornerstone of modern software development. In fact it is a real game changer—when Continuous Integration is introduced into an organization, it radically alters the way teams think about the whole development process. It has the potential to enable and trigger a series of incremental process improvements, going from a simple scheduled automated build right through to continuous delivery into production. A good CI infrastructure can streamline the development process right through to deployment, help detect and fix bugs faster, provide a useful project dashboard for both developers and non-developers, and ultimately, help teams deliver more real business value to the end user. Every professional development team, no matter how small, should be practicing CI. (Smart, 2011)

Below you will be able to see the continuous integration environment for jWebSocket projects.

1. Downloads process

Before install, configure and start the continuous integration environment it is necessary to download the following software.

* Apache Archiva (http://archiva.apache.org/)
* Jenkins (http://jenkins-ci.org/)
* Sonar (http://www.sonarsource.org/)

Continuous Integration Environment for jWebSocket is composed for three primary softwares.

Apache ArchivaTM is extensible repository management software that helps taking care of your own personal or enterprise-wide build artifact repository. It is the perfect companion for build tools such as Maven, Continuum, and ANT. (Archiva)

Jenkins, originally called Hudson, is an open source Continuous Integration tool written in Java. Boasting a dominant market share, Jenkins is used by teams of all sizes, for projects in a wide variety of languages and technologies, including .NET, Ruby, Groovy, Grails, PHP and more, as well as Java. (Smart, 2011)

Sonar is an open platform to manage code quality. As such, it covers the 7 aspects of code quality: (Sonar)

* Architecture and Design
* Duplications
* Unit Tests
* Complexity
* Potential bugs
* Coding rules
* Comments

1. Environment installation features.

Before to start the installations you need to have the personal computer with Ubuntu Server 11.10 64 bits with the next software’s installed.

* Apache2 server, latest version.
* MySQL server, latest version.
* Apache Tomcat6 server, or higher.
* Java 6 or higher.

1. Installation process

## Installing Subversion

Apache Subversion (often abbreviated SVN, after the command name svn) is a software versioning and revision control system distributed under an open source license. Developers use Subversion to maintain current and historical versions of files such as source code, web pages, and documentation. Its goal is to be a mostly-compatible successor to the widely used Concurrent Versions System (CVS). (Wikipedia)

Next you can see how to install Subversion server with your apache library and tools package, to do it is necessary execute this line at the console.

sudo apt-get install subversion libapache2-svn subversion-tools apache2-mpm-prefork

Please give the reader some hints how a successful installation looks like. What errors can appear? Can you provide a screenshot please?

## Installing Apache Archiva

The reader will ask “What is ‘Archiva’?”, so please give a sentence here.

**Creating the ARCHIVA\_HOME folder.**

To install Apache Archiva is necessary to create the ARCHIVA\_HOME folder; there you will locate all Archiva installation files. This folder will be created at our home folder.

mkdir /home/user/ARCHIVA\_HOME

The next step is to uncompress the latest Archiva version downloaded in this folder.

Finally we can start to the Archiva installations. To do so we need to open the ARCHIVA\_HOME and execute the following line at the console.

./archiva start

How can I see if all was successful? What kind of errors could appear and what do to then?

**3.3 Installing maven2**

Apache Maven is a software project management and comprehension tool. Based on the concept of a project object model (POM), Maven can manage a project's build, reporting and documentation from a central piece of information. (Apache Maven)

To install maven2 we only need to execute the following line at the console.

apt-get install maven2

**3.4 Installing Jenkins Server**

Again, some introduction desired! What is Jenkins, purpose? One sentence please.

Before start the Jenkins installation, we need to install some prerequisites like the daemon package. To do so we need to execute the following line at the console.

apt-get install daemon

Once installed the required tools we can start with the Jenkins installation. First it is necessary to open the directory where the Jenkins install file is located and start the installation. To do so we need to execute this line at the console.

dpkg -i jenkins\_1.448\_all.deb

**3.5 Installing Sonar Server**

Once more, please give some introduction sentence, what is Sonar about!

To install the Sonar server first we need to have the MySql server installed. To do it execute this line at the console:

Shouldn’t we have a separate paragraph then for MySQL, like for all other tools?

Apt-get install mysql-server

After the MySql server was installed, we need to create the sonar data base:

mysqladmin -p create sonar

Once the data base is created, we can start with the sonar installations. The first step is to download the latest version and uncompress it inSONAR\_HOME folder.

After that we need to edit the sonar configuration file (SONAR\_HOME/conf/sonar.properties) and put there the data base configurations. This file is easy to configure, we only need to comment the three lines where is configured the database embedded.

#DATABASE  
#Comment the 3 following lines to deactivate the default embedded database (used  
only for tests and demos)  
#sonar.jdbc.url:jdbc:derby://localhost:1527/sonar;create=true  
#sonar.jdbc.driverClassName:org.apache.derby.jdbc.ClientDriver  
#sonar.jdbc.validationQuery:values(1)

And uncomment the lines concerning to the MySql database.

#MySql  
#uncomment the 3 following lines to use MySQL  
sonar.jdbc.url: jdbc:mysql://localhost:3306/sonar?useUnicode=true&characterEncoding=utf8  
sonar.jdbc.driverClassName: com.mysql.jdbc.Driver  
sonar.jdbc.validationQuery: select 1

Then we need to put the configuration to connect with the sonar database created above. To do so, we need to find the “generic settings” and configure like this example:

generic settings  
sonar.jdbc.username: <your\_username>  
sonar.jdbc.password: <your\_password>

Once the parameters needed to the configuration are set, we proceed to generate the war to deploy it at our local tomcat server. To generate the war is necessary open this directory SONAR\_HOME/war and execute this line at the console.

./build-war.sh

This script is to generate a new .war file sonar application, then you can deploy it in any Tomcat server.

The next step is to install the Tomcat server. To do it execute the following line at the console.

Apt-get install tomcat6

Now we only need to copy it into our webapps tomcat folder (/ var/lib/tomcat6/webapps) and Tomcat server autodeploy the application automatily, then we can open the sonar server at this URL http://<your\_server\_ip>:8080/sonar

**4**. Configurations options.

**4.1 Configuring the Subversion Server**

The first step to configure the Subversion server is to create the svn folder in which your repositories are located.

mkdir /home/user/svn

Later the repositories will be created within this folder. To do so we only need to execute this line at the console:

svnadmin create /home/user/svn/jwebsocket

Later we need to create the repository folder structure, branches, tags and trunk:

svn mkdir --message="Setting up the directories..."   
file:///home/user/svn/jwebsocket/trunk   
file:///home/user/svn/jwebsocket/tags   
file:///home/user/svn/jwebsocket/branches

The next step is change the repository folder owner. We will put the user www-data as owner of the folder.

chown www-data:www-data /home/carlos/svn/jwebsocket/ -R

After we need to move the configuration file of /home/user/svn/jwebsocket.conf/authz to /home/user/svn/authz and edit it like in this example:

/etc/apache2/mods-avaiable/dav\_svn.conf  
<Location /svn>  
DAV svn  
#Repository folder  
SVNParentPath /home/carlos/svn/  
#Authentication mode  
#Name of the repository   
AuthName "Subversion Repository jWebSocket"  
  
AuthUserFile /etc/apache2/passwords   
# AuthBasicProvider ldap   
# AuthzLDAPAuthoritative on   
  
AuthzSVNAccessFile /home/user/svn/authz  
</Location>

Now you can enter to the repository using this URL http://<your\_ip>/svn/jwebsocket/.

Finally it is necessary to configure the subversion hooks. This hook is to allow subversion to execute the Jenkins remote compiling. To do so it is necessary to edit the file /home/usuario/svn/jwebsocket/hooks/post-commit.tmpl and add the following line at the file end.

# The first parameters is the SVN folder name and the second is the project Jenkins name  
/home/usuario/svn/jwebsocket/jenikins-launch-build.sh $REPOS $REV SVNFolderName JenkinsProjectName

Then we need to create the /home/usuario/svn/jwebsocket/jenikins-launch-build.sh file and append the following source.

#!/bin/bash  
# This script is executed after any subversion change,  
# and will notice the Jenkins server

REPOS="$1"  
REV="$2"  
PROJECT\_NAME="$3"  
JENKINS\_JOB="$4"

JENKINS\_USER=<Jenkins\_user>  
JENKINS\_PASSWORD=<jenkins\_password>  
JENKINS\_HOST=10.208.7.201:8002  
IS\_PROJECT\_CHANGED=`svnlook dirs-changed $REPOS --revision $REV | fgrep $PROJECT\_NAME`  
if [[ -n $IS\_PROJECT\_CHANGED ]]; then  
 wget --quiet --auth-no-challenge --no-check-certificate --http-user=$JENKINS\_USER --http-password=$JENKINS\_PASSWORD http://$JENKINS\_HOST/job/$JENKINS\_JOB/build?token=TOKEN  
 exit 0  
fi

**4.2 Configuring the Apache Archiva**

**Change the Archiva port to 8888**

Per Default, the Archiva server is using the 8080 port, the same of Tomcat. To avoid conflicts, it is necessary to change it to another port. To do so open the ARCHIVA\_HOME/conf/jetty.xml and find the <Set name="port"> label and change the default port of 8080 to 8888, here you have an example to this configuration.

<Set name="port"><SystemProperty name="jetty.port" default="8888"/></Set>

**4.3 Configuring maven2**

To configure the maven2 (why is this necessary, the reader can not know this here yet) is necessary edit the file /etc/maven2/settings.xml and change the mirrors. Following you can see an example for this file:

<mirror>  
 <id>archiva.default</id>  
 <url>http://<your\_archiva\_server\_ip>:8888/archiva/repository/internal/</url>  
 <mirrorOf>\*</mirrorOf>  
 </mirror>  
 <mirror>  
 <id>archiva.apache.snapshots</id>  
 <url>http://<your\_archiva\_server\_ip>:8888/archiva/repository/snapshots/</url>  
 <mirrorOf>apache.snapshots</mirrorOf>  
 </mirror>

This configuration is to use your local repository create using Apache Archiva.

**4.4 Configuring Jenkins Server**

Per default, Jenkins is using the 8080 port, this is the same port like Tomcat uses. Because of that it is necessary to change it to another port. To change the Jenkins port you need to edit the /etc/default/jenkins configuration file and change the default port of 8080 to 8002. Here you have an example:

# port for HTTP connector (default 8080; disable with -1)  
HTTP\_PORT=8002

After that you need to restart the Jenkins server:

/etc/init.d/jenkins restart

Now you can access the Jenkins server using this URL: http://<your\_server\_ip>8002/.

Finally you can proceed to configure plug-in to integrate Sonar Server (the reader does not know up to here what is “Sonar”, why does he need this “Sonar Server”? with Jenkins. To start it is necessary to configure the internet access in Jenkins, in this example we use the proxy server, to do so go to the administrator menu, plug-in settings, advance settings, then you can see the view to set the proxy configurations.



Picture 1.1 Example to proxy configuration in Jenkins.

Finally you can install the Sonar plug-in (“plug-in”? Above you mention a “Sonar server”. What is it about here?). To install it, go to the administration menu, plug-in settings, in the plug-ins tabs, find and select the sonar plug-in y press the “download new and install after restart” button.

**5. Administration of the applications.**

**5.1 Subversion administration.**

The first step of the subversion administration is to create the users. You can create a new Subversion user using the apache htpasswd command at the console. Only for the first time to create a user is use the –mc parameter, because the m parameter is to create the files where the users will be save, the other time only use the –c parameter. Now you can see an example to create the first user in Subversion.

htpasswd -mc /etc/apache2/passwords user

The other step is to set the security in subversion. To do so it is necessary to edit the file /home/usuario/svn/authz. The first step to configure the security is creating the groups. Following you can see an example to create a new group.

[groups]   
admins = user1, user2

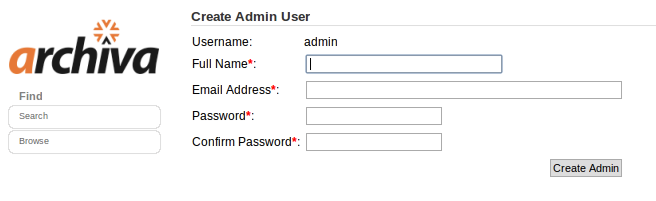
The next step is to assign privileges to the group. Here you can see an example how to do so.

[/]   
@admins = rw

How you can see, we assign read/write privileges to the admins group.

**5.2 Apache Archiva administration.**

When you enter into Apache Archiva the first time, Archiva shows you a view to create the administration user. Attached you see an example of this view.

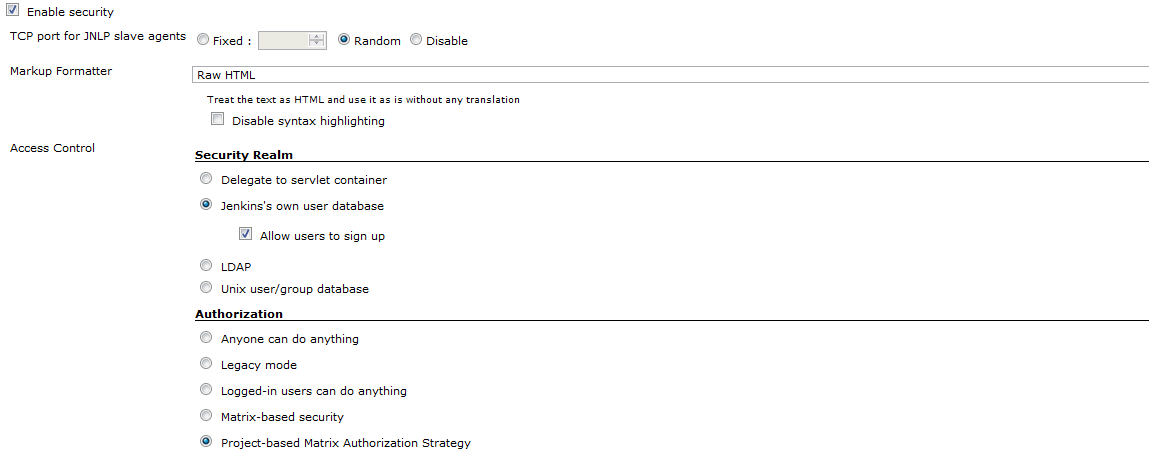


Picture 1.2 Form to create an administration user in Apache Archiva.

At the Apache Archiva, you only need one account, the administrator user, the other people can access without user.

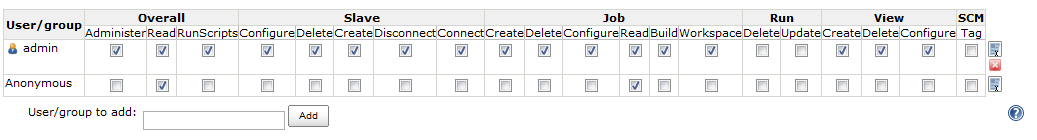
**5.3 Jenkins Administration.**

The first step to administer Jenkins is to apply security to the Jenkins server. Configuring security in Jenkins is very easy. Simply open the primary configuration page in Jenkins and activate the security by clicking at the “Enable security” checkbox. The screenshot shows an example how to enable the security in Jenkins Server.



Picture 1.3 How to enable the security in Jenkins server.

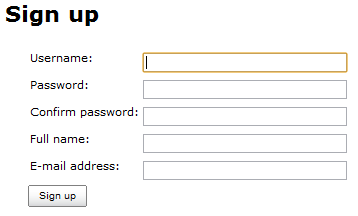
Following at the authorization session select the “strategy for the project security” and set the configuration like the following:



Picture 1.4 Example of strategy for the project security.

How you can see, we assign all privileges to the admin user and the anonymous user only has read permission.

The other step is create the admin user, to do it, click at the registration links located at the windows top. Following you can see the view with a form to create a new user.

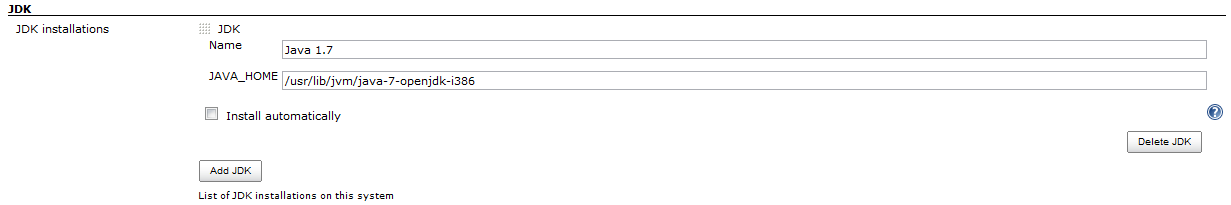


Picture 1.5 Creating a new user for Jenkins.

Can you please provide and english screenshot here?

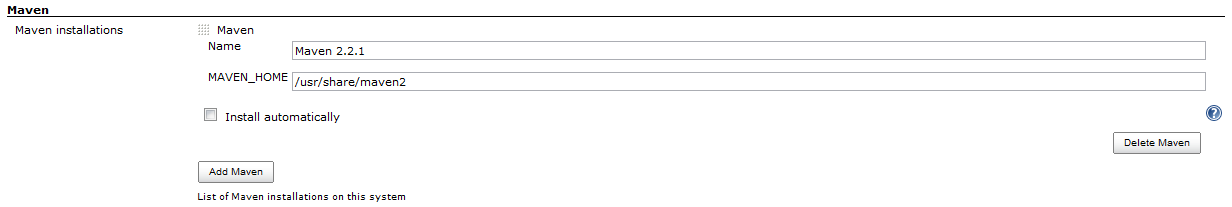
Another important step is to specify the JDK to be used by the Jenkins Server. To do so is necessary to open the primary settings page in Jenkins and go to the JDK section and type there where is our JAVA\_HOME. Following you can see an example of this configuration.

The Java Versions rupported are 6 or higher, in this case we use the openjdk 7.



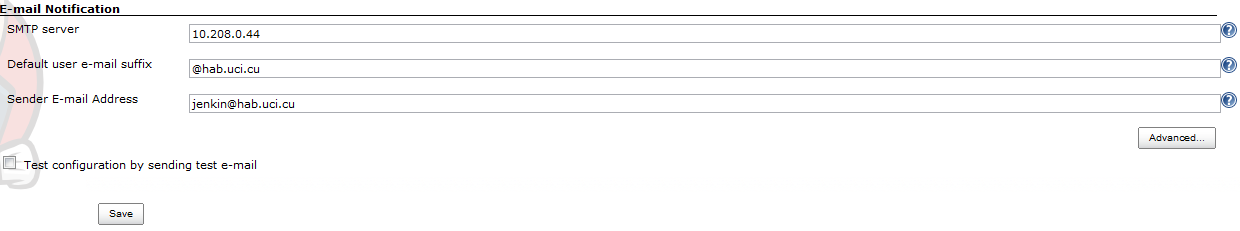
Picture 1.6 Configuring the JAVA\_HOME in Jenkins.

In the same way you configured Java, you need to configure your Maven version to be used. Go to the Maven section at the primary Jenkins settings page and type there the address to the MAVEN\_HOME. Following you can see an example how to do so.



Picture 1.7 Configuring the MAVEN\_HOME in Jenkins.

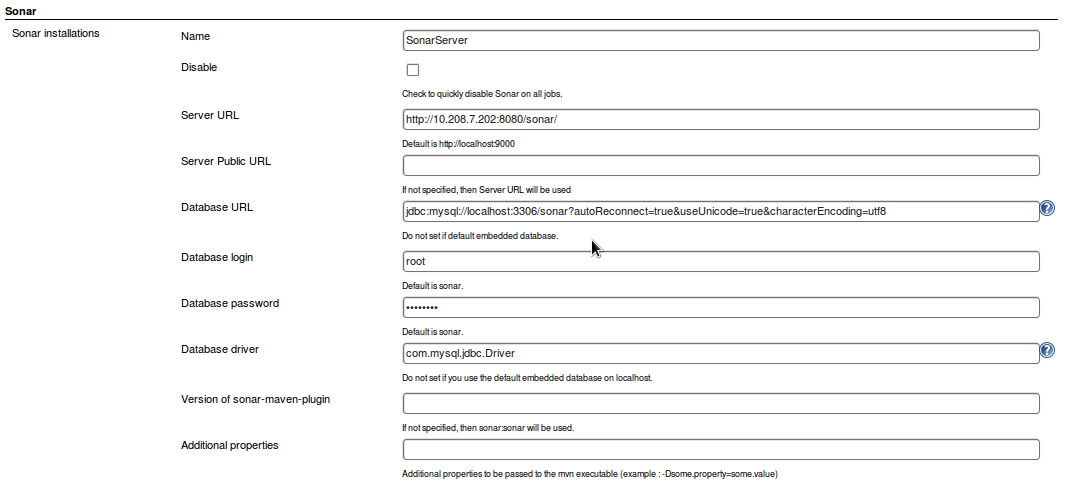
The next step is to configure the Jenkins email notifications. To do so it is necessary to go to the primary settings page in Jenkins, Notifications sections and put the same configurations like the following:



Picture 1.8 Configuring the email notification in Jenkins.

Another important step is to configure the sonar plug-in in Jenkins, this plug-in allows to integrate all Jenkins projects within the Sonar Server to get statistics like lines of code, comments, complexity, an others.

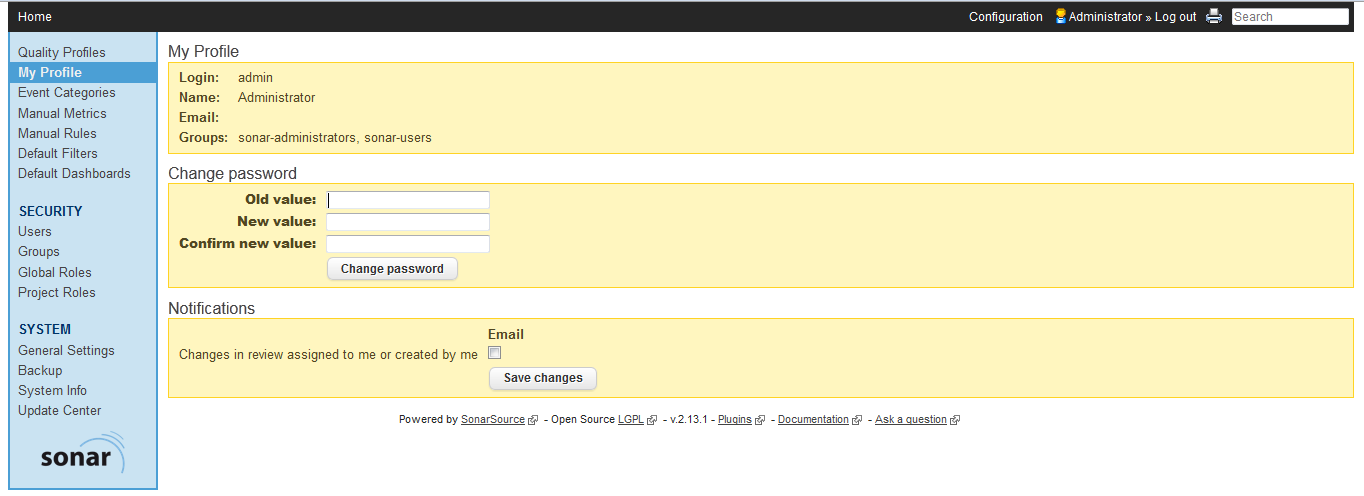
To set the Sonar plug-in configurations, go to the primary settings page at the Sonar section and type there all configurations like the following:



Picture 1.9 Configuring the Sonar Plug-in in Jenkins.

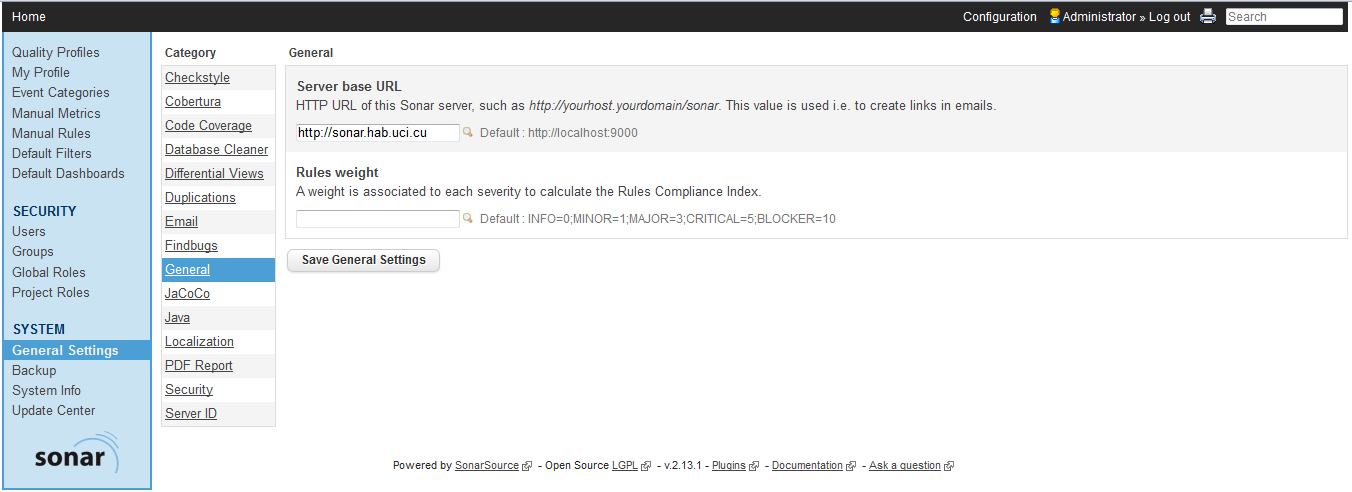
**5.4 Sonar Administration.**

Once the sonar server is installed, the first step to do is to change the default password. When you install sonar, it has by default an administration account created with this credentials (user: admin and password: admin)To change the administration password is necessary start a session in sonar using the administration account created by default(admin/admin). Then click at the administration link to change the admin profile and change the administration password.



Picture 1.10 Change the Sonar admin profile.

The next step is to set the URL where the Sonar server is located. To do so is necessary to go to the settings menu, general settings, general, and then you can see the form to set the Server base URL.



Picture 1.11 Primary sonar settings.

The next step is to activate the PDF report plug-in. The first step to do is to configure the internet connection. To do so, we need to edit the /home/usuario/SONAR\_HOME/conf/sonar.properties file and set the following configuration:

#---------------------------------------------------------  
# UPDATE CENTER  
#---------------------------------------------------------

# The Update Center requires an internet connection to request http://update.sonarsource.org

# It is activated by default:

#sonar.updatecenter.activate=true

# HTTP proxy (default none)

http.proxyHost=<yourhost\_ip\_number>

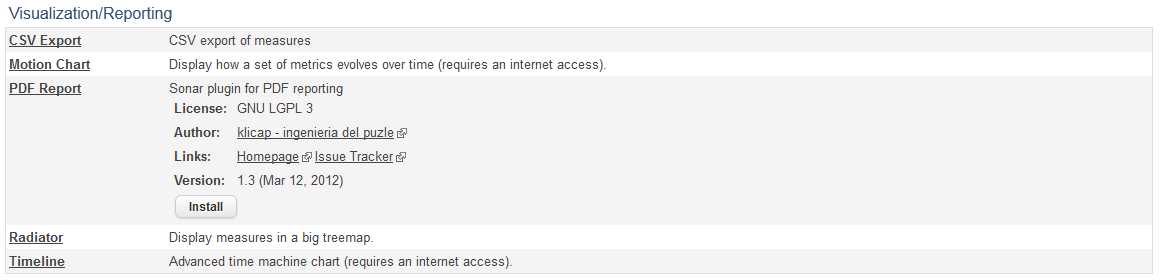
http.proxyPort=3128

# NT domain name if NTLM proxy is used

#http.auth.ntlm.domain=

# SOCKS proxy (default none)  
#socksProxyHost=  
#socksProxyPort=  
# proxy authentication. The 2 following properties are used for HTTP and SOCKS proxies.  
http.proxyUser=usuario  
http.proxyPassword=<your password>

Ultimately you can install the PDF report plug-in. This plug-in generates a pdf file with some statistics like lines of code, how many package, class, methods you have, number of comments, complexity of your code and coding rules violations per class. To install this plug-in, go to the settings menu, update center, and click at the plug-in tab, find the PDF Report plug-in and install it.



Picture 1.12. Installing PDF report plug-in in Sonar