**Crossover Application**

**Summary**

1. Introduction
2. Applications
3. Requirements
4. Build and Deploy
5. Architecture
6. Notes
7. **Introduction**

I am Marcos, 31 years old and I wish very much this job so I can have an abroad job experience, learn new cultures, work, organization, etc. I am pursuing that desire for a long time, I have 9 years of experience with Java development and I think this is the best opportunity to take. In my everyday life, I like to play football, practice triathlon, study and always be update with the world.

**2) Applications**

There were three applications developed for the test: client (crossover-client), server (crossover-server) and log server (log-server). I will explain the basics of each one:

Client (crossover-client): It is a web application where the main features are:

* Login in the application
* Upload and compile a source code
* Check the compilation results.

Server (crossover-server): It is a services based application, It provides services to receive, compile and deliver the results of the compilation of the submitted source code.

Log Server (log-server): It is a central log server responsible to receive log messages from both Client and Server applications and write to a file (application-error.log).

**3- Requirements**

* Java 1.7.0\_79 (Configure environment variables: JAVA\_HOME). Download <http://www.oracle.com/technetwork/java/javase/downloads/index.html>

* Maven 3.3.3 (Configure environment variables: MAVEN\_HOME). Download <https://maven.apache.org/download.cgi>
* JBoss EAP 6.4 (Configure environment variables: JBOSS\_HOME). Download <http://jbossas.jboss.org/downloads/>
* Read and write permission on the program files including directories and subdirectories.
* Access to the Maven repositories.

**4- Build and Deploy**

The steps to build and execute the applications should be the same on Linux and Windows environment, so let’s do the steps below:

1. Extract the zip file
2. Access the folder <Code>/crossover-client
3. Execute the Maven command:

mvn package -U

4) Copy the generated WAR file to the JBoss Standalone deploy folder

Copy the file:

<Code>/crossover-client/target/crossover-client-0.0.1-SNAPSHOT

To:

<JBOSS\_HOME>/standalone/deployments

5) Access the folder <Code>/crossover-server

6)Execute the Maven command:

mvn package -U

7) Copy the generated WAR file to the JBoss Standalone deploy folder

Copy the file:

<Code>/crossover-server/target/crossover-server-0.0.1-SNAPSHOT

To:

<JBOSS\_HOME>/standalone/deployments

8) Start the JBOSS server:

Windows: <JBOSS\_HOME>/bin/standalone.bat

Linux: <JBOSS\_HOME>/bin/standalone.sh

8) Access the folder <Code>/log-server

9)Execute the Maven command:

mvn install

10) Access the folder

<Code>/log-server/target/

11) Execute the command:

java -jar log-server-0.0.1-SNAPSHOT-jar-with-dependencies.jar 4712 ../log4j-server.properties

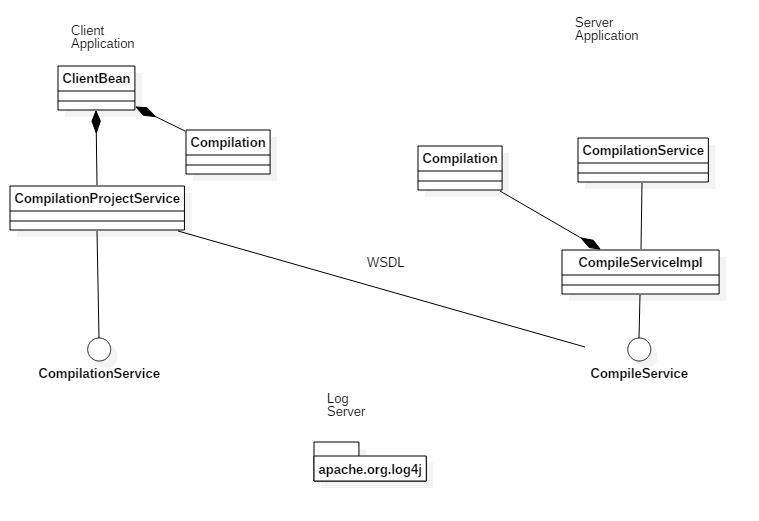
Note: if the port number “4712” is not available, it will be necessary to alter that configuration. The configuration file “log4j.properties” of the applications should be altered for an available port number.

**5) Architecture**

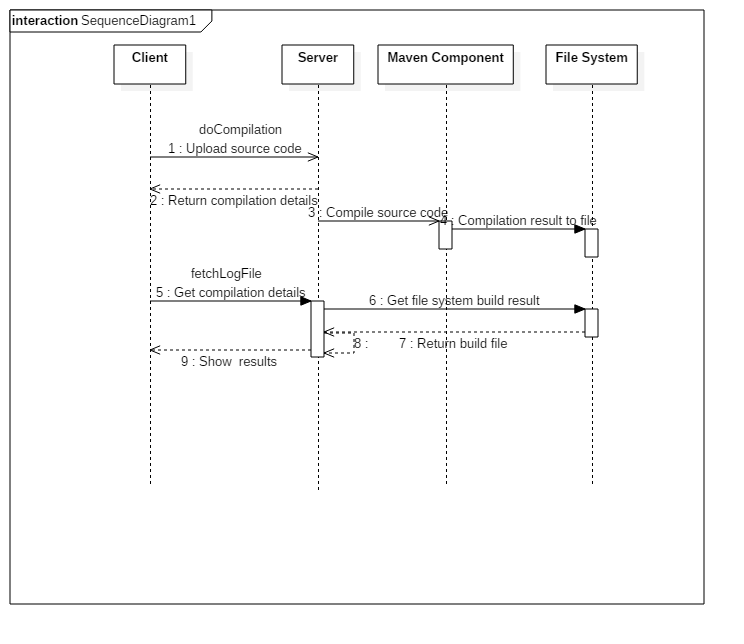
The main technologies and framework involved in the development of the solution were:

* Java 7.0.x: To be able to used other modern frameworks like JSF 2.2
* JBoss 6.4 EAP: web container with support to JSF 2.2
* Spring framework (Spring Security, Dependency Injection)
* Apache CXF 3.1.1: for the SOAP web services development.
* JSF 2.2: To get productivity on visual interface development and follow the MVC design pattern as well as get modularity, component based and reusability.

UML diagrams:

Classes diagram

Sequence diagram



**6) Notes**

The applications development were a demonstration of my knowledge of the most used technologies and approaches for software development. I have tried to make them simple because of restrictions of the project involving quality, scope and time.

The Client application communicates with the server through SOAP protocol transferring binary zip file with the source code. It was Used CXF framework to develop the service provider as well as the service consumer.There are 2 web services: One to send the zip file with the source code and other to fetch the result of the compilation.

Consideration have to be made about security, since it was not a requirement, the transfer have no cryptography, no digital certificate and no other security mechanism. The only security requirement was about authentication and authorization, it was implemented using Spring Security.

The applications do not use Database, i.e, the artifacts like compilation log files are stored on the file system (server), so the program must have read and write permission on the folder where the application will be run.

Note: The client application is in browser session scope, so when the browser session expires, all data are lost.

The Log Server was implemented using the Log4J framework, it opens a socket listening to a port number (4712). If there is issue regarding availability on port “4712” changes on the log4j.properties file of the applications must be made. The log file is located at <log-server>/target/application-error.log.

It was used Maven to to build the application and compile the sample application. The sample source codes are located in the folder:

<crossover-client>/sample/crossover-client.zip and <crossover-server>/sample/crossover-client.zip

They can be used to execute server and client unit tests and also to be used as the source code upload in the client application. The client application is accessible through the URL

<http://localhost:8080/crossover-client>

Use the credentials:

User: ADMIN

Password: ADMINPASS

The web service descriptor (WSDL) of the server application is:

<http://localhost:8080/crossover-server/services/compile?wsdl>