**Efficient Task Scheduler for Distributed Systems - Simulator Design Document**

Group members: (Name:ID)

Joshua Archer: 45235074

Wei Xuan Leow: 44317484

**Introduction:**

This stage of the project is focused on developing a simple job dispatcher via a client. This client will connect to a server which contains multiple server types dispatching different jobs, ordered appropriately. The client connects to the server through an ip address and a port number on that device. Once connected, a set of connection messages will be displayed on the client and then the scheduling of jobs can be done.

**System overview:**

The client system was developed using Java since both of the group members proficiency in it, as well as the task being suited to an object-oriented programming language.

This system schedules requests and sends them over to the server side for processing. Communication is continuous between both the server and the client throughout the whole stage acknowledging when connection has been made/lost and every time a request has been carried out, along with some extra data like how long it took and how expensive it was with some more details. The client is responsible for sending jobs to different server types. These server types are ordered by first of the largest server first.

The design for this system is using an agile approach in terms of our adaptive development approach. Since the group two work separately most of the time, this is seen as the best approach and having an adaptive solution to solve the problems as they approach. The system includes extra methods for the need of future development or scalability.

Constraint: Time limited the depth of design.

**System Architecture:**

The client system includes multiple features, a class for; connecting the client to the server, parsing XML documents using the SAX-based library into the server and then displaying the information in the correct format.

Hardware- 1 computer with various resources.

Software- Virtual box (simulating Linux), client application, server application, Java-8, Sublime text editor.

User interfacing- A human needs to; start the connection between client and server, change port and ip address if need be, open file to parse xml files, quit the program.

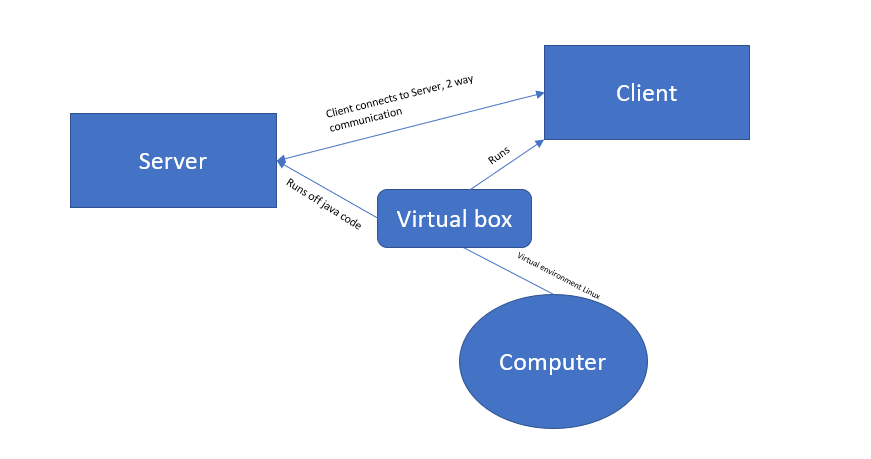


Image 1: Connection between hardware/software

**Implementation Details:**

Code development was made in text editors and transfers of code was sent along via email and also posted in a Bitbucket account (link below). Documentations were also shared via the Bitbucket. Java was the chosen language due to both being proficient in this and Virtual boxes were used to simulate Linux.

Libraries used include the following; *org.w3c.dom,* *javax.xml.parsers and java.io.*

Data structures for connecting the client to the server include an object-oriented approach, to store XML files in the server were NodeLists and to parse was using javax XML parser.

Team responsibilities include; Joshua was in charge of the Simulator Design Document and

Wei Xuan developed most of the code due to Josh having troubleshooting issues.

**References:**

Shared bitbucket for code and resource management: <https://bitbucket.org/joshua15/comp335_group12/src/master/>

[1] “Socket programming in java”, GeeksForGeeks. [Accessed 01/04/19]

<https://www.geeksforgeeks.org/socket-programming-in-java/>