**Distributed Systems Job Scheduler Using Largest Available Server**

Luke Glover

Jarrod Adair

Frederick Jack

**Introduction**

Introduction (½ page): What this project (focusing on Stage 1) is about, including the goal of the project and Stage 1.

In a distributed system job allocation is very important for the efficient use of resources. Scheduling is a decision-making process that typically uses an optimizing algorithm for dispatching of jobs. This project is implementing a client-side job scheduler working with a distributed systems server simulator. The task of Stage 1 of the project is to schedule and dispatch every job to the largest server available in each simulation. The client is connected through a socket, so it is language independent, but this implementation is written in Java. The goal of this stage of the project is to successfully run and scheduled all available jobs without failures.

**System Overview**

System overview (½ page): high-level description of the system (both client-side simulator and server-side simulator with the focus being your client-side simulator), preferably, with a figure (your own, not one in ds-sim User Guide) showing the workflow/working of the system.

At a high-level, the distributed systems server (ds-server) is started using the correct command line prompt. At that point, the client is started with the correct command line prompt which will then create a connection. The server then reads the configuration file specified and writes an XML file with the servers available. The client side will read the XML file to find out the servers that are available. The client-side simulator is a large loop that cycles through protocol states. The steps are that the ds-server generates a job and sends it to the client for scheduling. The client receives the new job with details. The client assigns/schedules the job to the largest server and the process repeats until there are no new jobs. The client then sends a quit command to the server and closes the connection.

![Diagram

Description automatically generated]()

**Design**

Design (1 page): design philosophy, considerations and constraints, functionalities

of each simulator component focusing on the client-side simulator.

**Implementation**

Implementation (2 pages): brief description of any implementation specific information including technologies, techniques, software libraries and data.

structures used. How each of components/functions of your simulator is implemented including who oversees which function(s) and how they have led the design and development.

References including project git repository/wiki, e.g., GitHub and Bitbucket.

\* The numbers of pages for each section are also a suggestion.