**A Distributed System's Task Scheduler**

**Group – 04**

**GitHub URL:** https://github.com/PieInOblivion/COMP3100\_STAGE1

**Student Name:** Lucas Hofmockel-Spanakis

**Student ID:** 45422745

**GitHub URL:** https://github.com/PieInOblivion

**Student Name:** Muhammad Khaled

**Student ID:** 44978286

**GitHub URL:** https://github.com/MuhammadKhaled786

**Student Name:** Saidul Islam Kabboy

**Student ID:** 45669147

This project aims to identify the largest server in the stimulated distributed system and to develop a scheduling system on the distributed system's client end to assign the job to the server. Scheduling is a crucial method for ensuring the successful operation of computing networks, particularly distributed systems. Scheduling is fundamentally a decision-making mechanism that allows for resource sharing among a variety of operations by deciding their execution order based on the available resources. The introduction of distributed networks, such as clusters and grids, introduced new scheduling problems in computer systems. Task allocation is the method of allocating tasks to the system's appropriate processors, while task scheduling is used to evaluate the order in which tasks are executed. In certain instances, task scheduling incorporates the allocation procedure as well.

The stimulated distributed system is built with two sides in view: the client and the server. The server-side simulator has already been developed and made available. It is in control of the simulation. It specifically simulates a distributed system of user-specified configurations (e.g., ds-config01.xml) and collaborates with the client-side simulator. The server-side simulator's IP address and default port number are 127.0.0.1 and 50000, respectively. The client side of the stimulated distributed system is the focus of this project. The client side is supposed to be configured in such a way that it can first establish a connection to the servers, then define the largest available server by reading the configuration from the given xml file, and afterwards retrieve data and schedule the jobs to the largest available server.

In the stimulated distributed system, the client class acts as the client side. There are two instances of the client class.

1. The Server instance: The Server instance had the server attribute as the instance of the server instance and a Server constructor for all the instances.
2. The Jobn instance: the Jobn instance had the job attribute as the instance of the job instance and a Jobn constructor with all the instances.

Many methods are also available on the client side of the stimulated distributed system.

1. The first is the ServerConnection() method, which assists the client side in establishing a connection to the distributed system's server side. The ServerConnection() method has 5 methods for managing the connection.
   1. The ServerConnection method, which is used to connect to the server by exchanging messages.
   2. The send method sends messages to the server in order to request a connection and confirm authentication.
   3. The read method reads the response from the server and replies.
   4. The flush method removes all the server's previous tasks.
   5. The close method terminates the connection between the client and the server.
2. The importXML() method has many components that are responsible for reading and preserving the server's configuration.
   1. The first phase of the method attempts to locate and normalize the file containing the configuration of the distributed system's server.
   2. The configurations of the servers from the directory are retained in an arraylist of server instance data type in the method's next segment.
3. The sortServerList() method compares the core count of the servers with the sort and compare method from the java library to sort the servers in ascending order. If there are multiple servers with the same core count, the type is used as a secondary parameter.
4. There are several processes in the main() approach as well.
   1. The serverConnection method, which is the first phase of the main method, is used to connect to the distributed system's server. It also sends the necessary message, reads the response, and responds, establishing a connection.
   2. The subsequent process read the xml file and build an arraylist of servers based on the xml file's configuration. The array list is then sorted in ascending order.
   3. The following process retrieves jobs from the server via the BufferedReader initialised through the read() method and schedules the job to the largest server available, using the allToLargest() function.
   4. Finally, to flush the server and close the connection, the main() method calls the flush and close method once the server responds that there are no more jobs to schedule.