Program 2 Result report

Joshua Landron

CSS434 Spring19

Table of Contents

Algorithm Documentation2

Figure 1.1: UnixClient Overview and decision flow 2

Figure 1.2: UnixAgent Overview and decision flow 3

Best execution times for UnixClient and UnixAgent4

Figure 2.1: UnixClient #nodes #commands (who ls ps df) 4

Figure 2.2: UnixClient #nodes 1 command (grep\ -o\ 123\ ../files/text1.txt) 4

Figure 2.3: UnixClient #nodes 1 command (cat\ ../files/text1.txt | grep -o 123 | wc -l) 4

Figure 2.4: UnixAgent #nodes #commands (who ls ps df) 4

Figure 2.5: UnixAgent #nodes 1 command (grep\ -o\ 123\ ../files/text1.txt) 4

Discussion5

Programmability5

Performance comparison 5

# Algorithm Documentation:

Figure 1.1 shows the flow of operation for the UnixClient. This client uses RMI to remotely execute terminal commands on UnixServers. It starts by checking if there were enough arguments passed into it to initialize the printing value, the servers it will connect to, and the commands to be remotely executed. The UnixClient then iterates through args and initializes all the needed values and connecting to the indicated servers. It then begins the main execution loop, iterating through connected servers and executing the given commands.

Figure 1.1: UnixClient Overview and decision flow

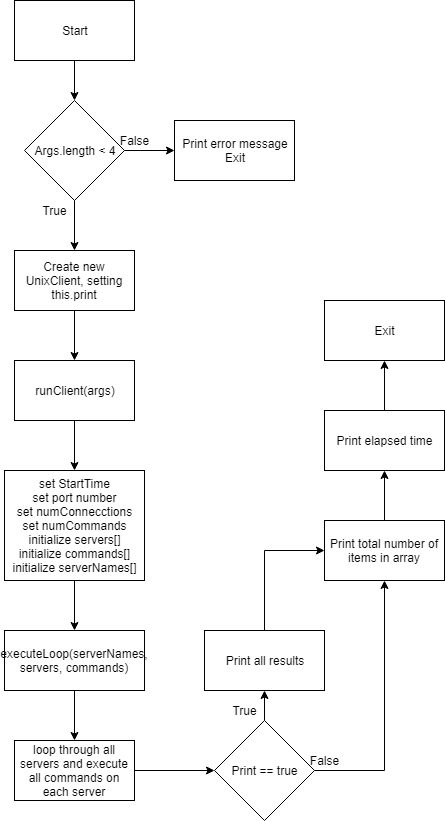
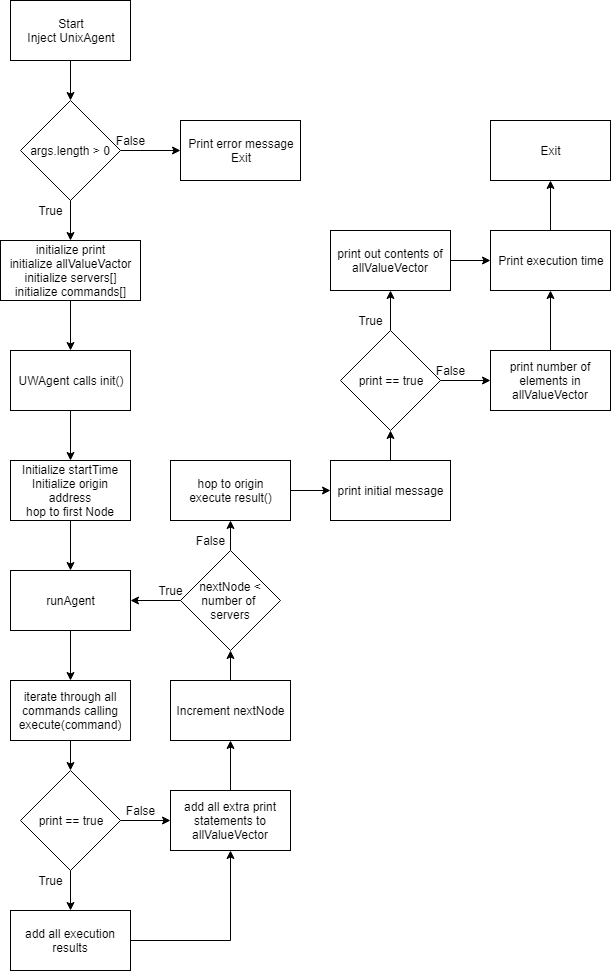


Figure 1.2 shows the flow of operation for the UnixAgent. This client uses UWAgent to migrate among a list of computers and execute a list of commands on each computer. It starts by checking if there were enough arguments passed into it to initialize the printing value, the servers it will migrate to, and the commands to be executed. The UnixAgent then hops to the computers in order and iterates through the given commands while saving all of the results in a vector that will be maintained throughout the life of the agent. Once the agent has visited all computers, it will hop back to the original computer and print the requested output.

Figure 1.2: UnixAgent Overview and decision flow



# Best execution times for UnixClient and UnixAgent:

## Figure 2.1: UnixClient #nodes #commands (who ls ps df)

|  |  |  |
| --- | --- | --- |
| #nodes | 4 commands | 12 commands |
| 1 | 94 | 151 |
| 2 | 134 | 238 |
| 3 | 175 | 319 |

Figure 2.2: UnixClient #nodes 1 command (grep\ -o\ 123\ ../files/text1.txt)

|  |  |
| --- | --- |
| #nodes |  |
| 1 | 171 |
| 2 | 284 |
| 3 | 392 |

## Figure 2.3: UnixClient #nodes 1 command (cat\ ../files/text1.txt | grep -o 123 | wc -l)

|  |  |
| --- | --- |
| #nodes |  |
| 1 | 2419 |
| 2 | 4932 |
| 3 | 7784 |

Figure 2.4: UnixAgent #nodes #commands (who ls ps df)

|  |  |  |
| --- | --- | --- |
| #nodes | 4 commands | 12 commands |
| 1 | 47 | 89 |
| 2 | 76 | 158 |
| 3 | 92 | 220 |

Figure 2.5: UnixAgent #nodes 1 command (grep\ -o\ 123\ ../files/text1.txt)

|  |  |
| --- | --- |
| #nodes |  |
| 1 | 116 |
| 2 | 226 |
| 3 | 325 |

Complete execution results as well as my shell scripts to carry out each command 5 times in order are included in the files with the source code.

# Discussion:

**Programmability**

The total number of lines of code in the UnixClient, UnixServer, and ServerInterface is about 150 lines total, in three classes. This is somewhat simple to understand, as the code is executed from a single client on multiple servers. The total number of lines of code in UnixAgent is about 100, and all in one class. This is simpler to code, but difficult to understand on the first pass. The idea that the class is moving from one PC node to another is different than anything else we have done. This change was not to difficult to implement after a bit of basic outlining of the specifications of a java Agent though.

**Performance comparison**

The performance of the UnixAgent was consistently faster than the UnixClient. This is due to the simpler communication structure that exists when the agent fully migrates to a computer and is able to execute all commands while at that computing node directly and add the results the its vector without any messages being passed over a network. The UnixClient must open a line of communication with a server, order the execution of a command, and receive the results. This means that for 4 commands, there are 8 communications over the network, while the Agent would have only 2 contact points with the network, migrating to and from the computer.