

Software Architectures

Practical Assignment 3 Report

Leandro Silva, 93446
Margarida Martins, 93169

Monitor

The Monitor process is responsible for knowing the status of the system.

The Monitor implements a heartbeat system in order to see if some server or load balancer went down and act accordingly. Firstly the monitor sends a heartbeat message to the respective service/load balancer and waits the amount of time defined in its initial configuration. After waiting it will consider every process as down if it did not respond to the message. When the monitor detects that the primary loadBalancer died it will send a message to the secondary so that it knows he has to take the primary place. When the monitor detects that a server is down it will send all the requests it was processing to the loadBalancer.

Monitor

Port Number: 5000
Heatbeat Thres...: 5000
Load Balancer Status: UP

Load Balancer | Server | Clients

Request ID	Client ID	Server ID	IN	DeadLine
1	5003	5002	5	10
2	5003	5002	2	10
3	5003	5002	2	10
4	5003	5002	2	10
5	5003	5002	5	10

Terminate

Monitor Creator

Port: 5 000

Heartbeat Threshold (ms): 5 000

Start Monitor

Monitor GUI's

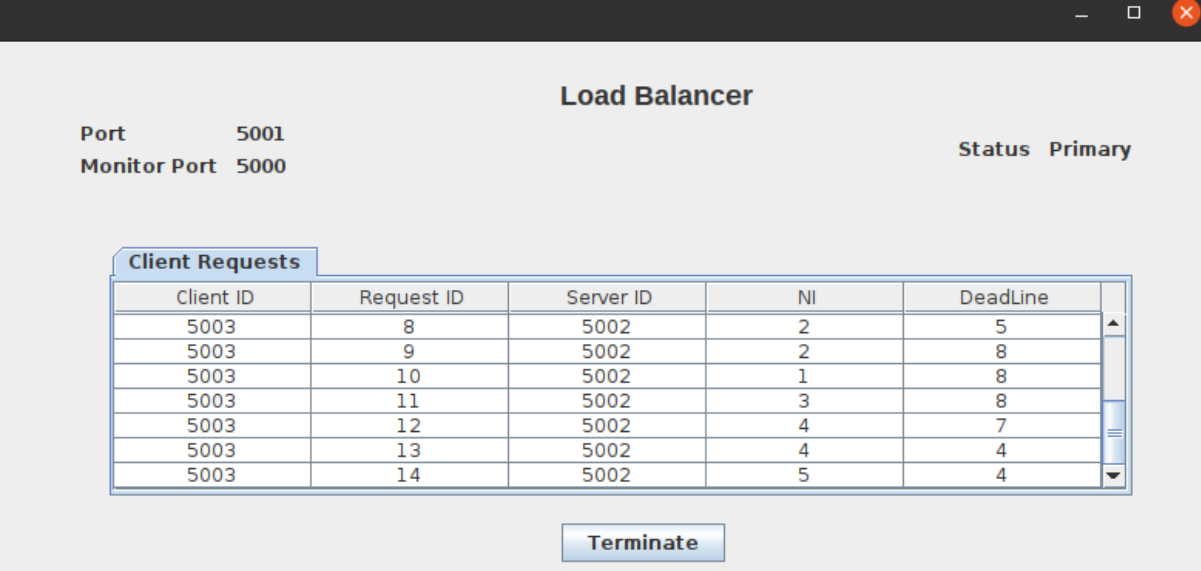
Load Balancer

The Load Balancer is responsible to receive Client requests and forward the requests to the best server.

The calculation of the best server is done using the current system status, for that the load balancer asks the monitor in order to know the current active servers and the number of requests (processing and pending) and the total number of iterations each has. The server with the lowest number of requests is always chosen, if there is more than one server the one with the smallest number of iterations will be chosen.

The secondary load balancer will be waiting until it receives a message from the monitor in order to become the primary one.

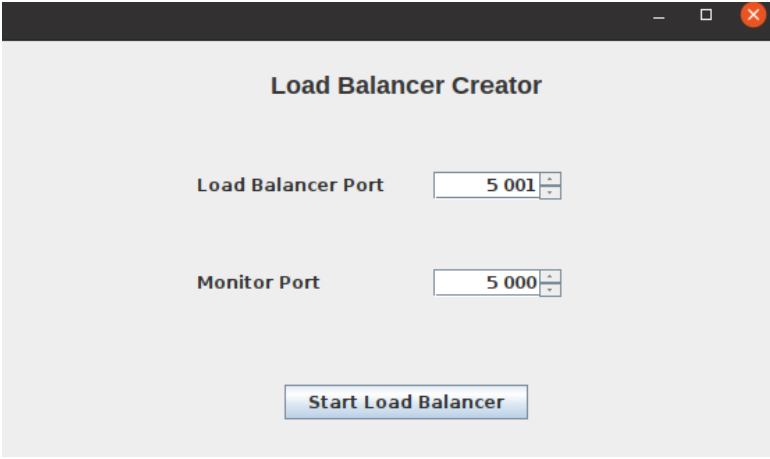
In order to make client-load balancer connection the most transparent as possible, when the primary load balancer dies, the secondary will connect to the client in the same port as the primary one.



The screenshot shows a window titled "Load Balancer". At the top left, it displays "Port 5001" and "Monitor Port 5000". At the top right, it shows "Status Primary". Below this is a tab labeled "Client Requests" which contains a table with the following data:

Client ID	Request ID	Server ID	NI	DeadLine
5003	8	5002	2	5
5003	9	5002	2	8
5003	10	5002	1	8
5003	11	5002	3	8
5003	12	5002	4	7
5003	13	5002	4	4
5003	14	5002	5	4

Below the table is a "Terminate" button.



The screenshot shows a window titled "Load Balancer Creator". It contains two input fields: "Load Balancer Port" with the value "5 001" and "Monitor Port" with the value "5 000". Both fields have increment and decrement buttons. At the bottom is a "Start Load Balancer" button.

Load Balancer GUI's

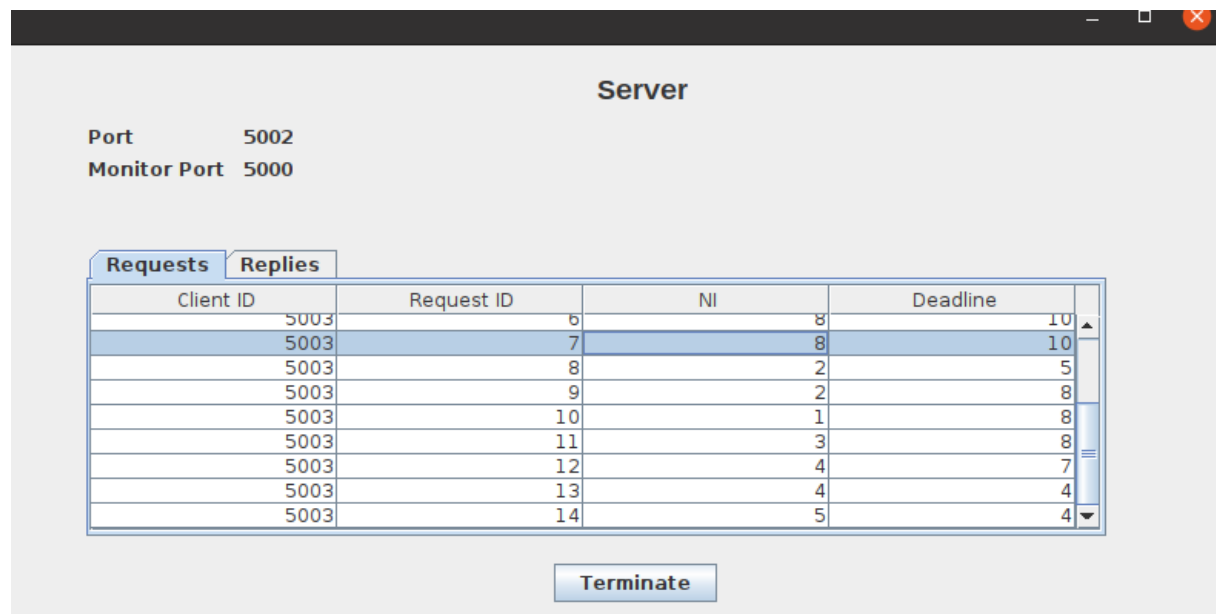
Server

The Server is responsible to receive requests, make the calculations and reply to the respective client.

When receiving a request the server checks whether he is capable of accepting it (if the total number of requests is less than five or the total number of iterations is less than twenty).

Each server always has three worker threads running. These threads will be awakened when there are calculations to be done.

When there are two pending requests and there is a free worker thread the request that will be picked will be the one with the smallest deadline.



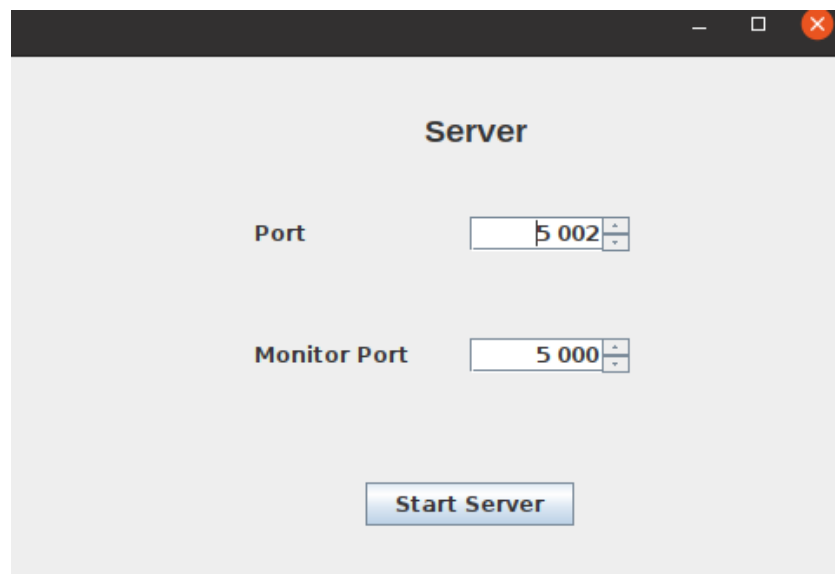
The screenshot shows a window titled "Server". It displays the following configuration:

- Port: 5002
- Monitor Port: 5000

Below the configuration, there are two tabs: "Requests" (selected) and "Replies". The "Requests" tab contains a table with the following data:

Client ID	Request ID	NI	Deadline
5003	6	8	10
5003	7	8	10
5003	8	2	5
5003	9	2	8
5003	10	1	8
5003	11	3	8
5003	12	4	7
5003	13	4	4
5003	14	5	4

At the bottom of the window, there is a "Terminate" button.



The screenshot shows a window titled "Server". It displays the following configuration:

- Port: 5002
- Monitor Port: 5000

At the bottom of the window, there is a "Start Server" button.

Server GUIs

Client

The client is responsible to make requests to the load balancer and receive replies from a server.

The client can send as many requests as he wishes which can then be rejected or accepted by a server.

The requests can have different number of iterations and deadlines.

The screenshot shows a window titled "Client" with a light gray background. At the top, it displays "Client Port Number 5003" and "Load Balancer Port Number 5001". Below this is a section titled "Requests" containing a table with three rows of request data. To the right of the requests table is a "Send new Request" panel with input fields for "Number of iterations" (set to 1) and "Deadline" (set to 7), and a "Send Request" button. Below the requests section is a "Replies" section with a table showing two rows of reply data. At the bottom center of the window is a "Terminate" button.

Request Id	Number of Iterations	Deadline
1	7	10
2	1	10
3	1	7

Request Id	Server Id	Number of Iterations	Deadline	Pi
2	5002	1	10	3.1
3	5002	1	7	3.1

Client GUI showing Requests and Replies.

The screenshot shows a window titled "Client" with a light gray background. It contains two input fields: "Client Port Number" with the value "5 003" and "Load Balancer Port Number" with the value "5 001". Both input fields have small up/down arrow buttons on their right sides. At the bottom center of the window is a "Start Client" button.

Client GUI showing input fields for Client Port Number (5 003) and Load Balancer Port Number (5 001), and a Start Client button.

Client GUIs

Features not Implemented

In this assignment we were able to implement all the use cases and its respective features.

Contribution

Leandro Silva - 50%

Margarida Martins - 50%