LAZYBOT – SERVERS COMMUNICATION

This documentation is made to understand the communication between the microservices.

To use the project and learn more about it, go to this Github repository:
https://github.com/Ronflonflon/lazybot

Summary

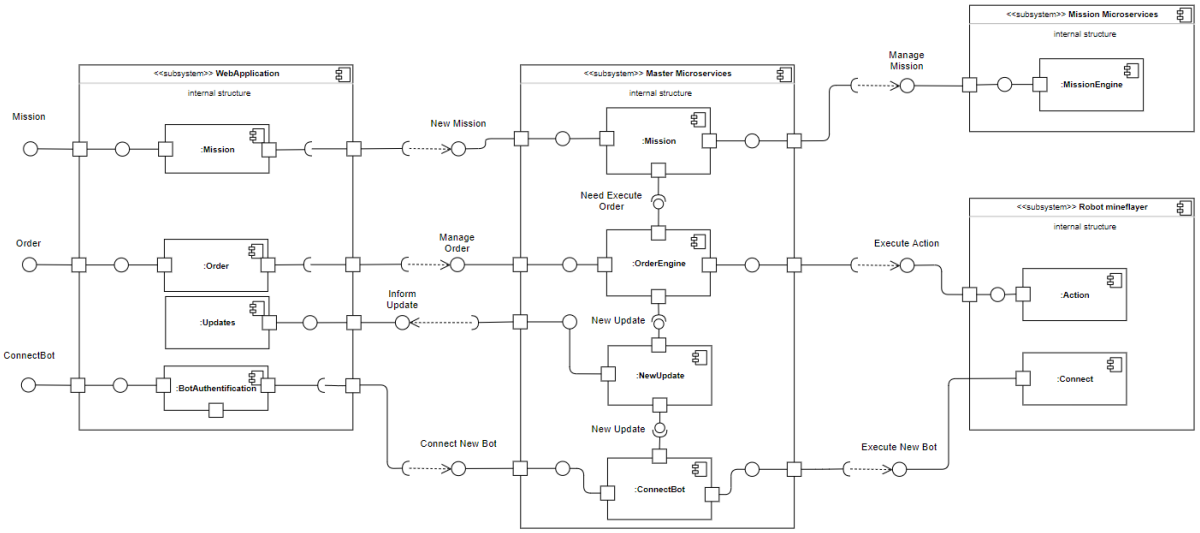
1. Introduction

Lazybot is made to help test the performance of Minecraft servers.

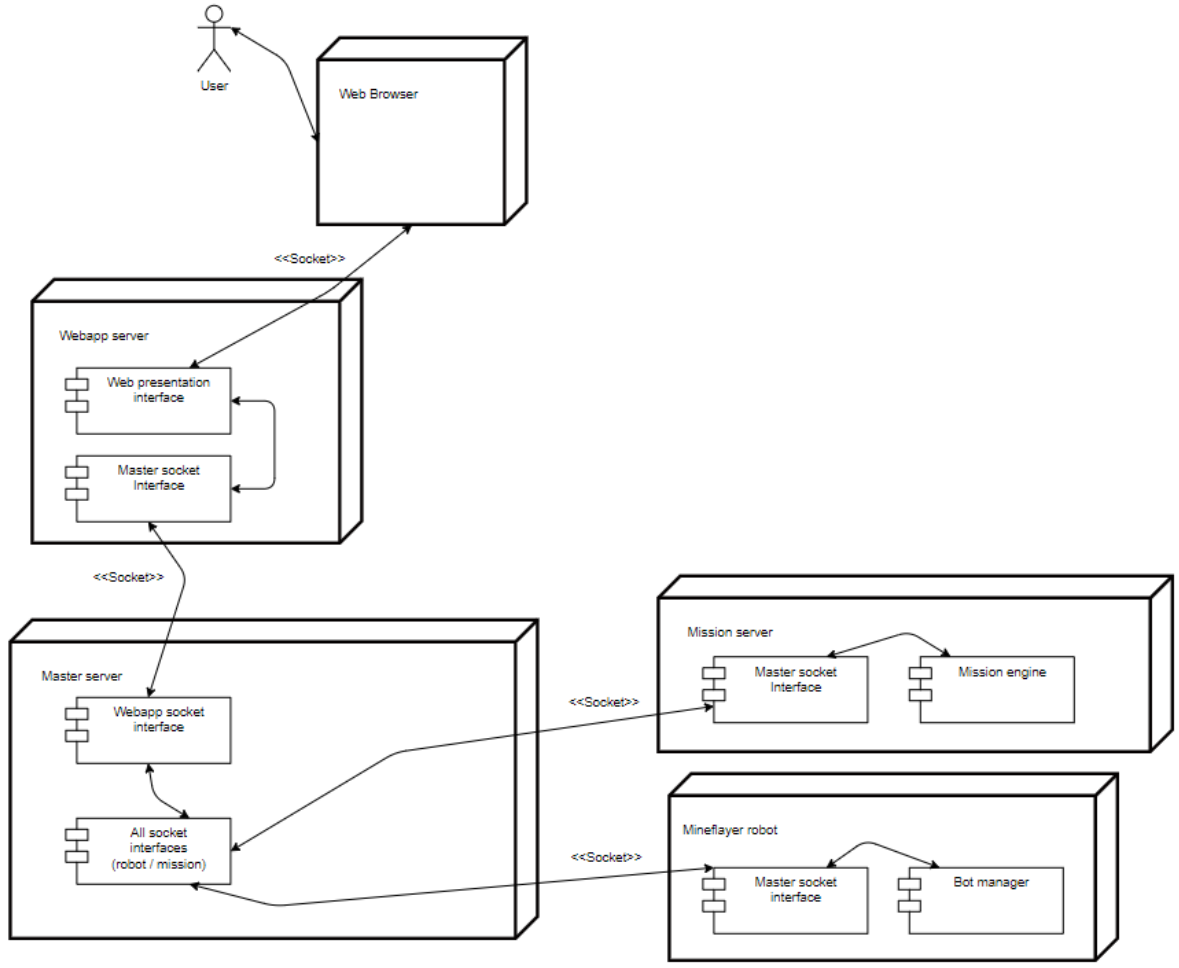
In the first version, Lazybot is made with 4 microservices :

- lazybot-webapp for the web interface ;
- lazybot-master to link the communication between all the microservices ;
- lazybot-mission done to compose missions that robots will then have to do ;
- lazybot-map to do some heavy calculations on the map.

2. Component diagram



3. Deployment diagram

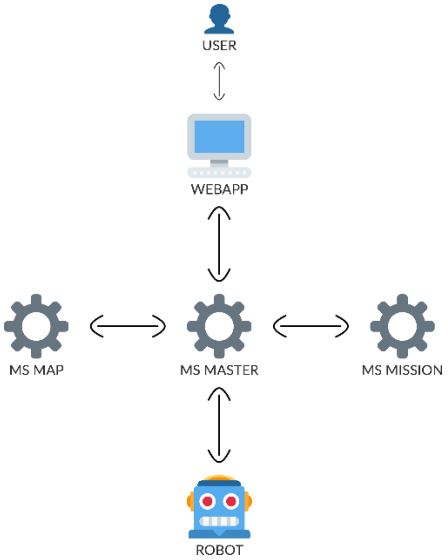


4. Microservice communication

The communication between the microservices work with the Sockets in JSON format.

The entire documentation below is also present in the source code of the project with the Javadoc.

To easly understand which microservice communicates with whom, have a look to the following schema :



4.1 lazybot-webapp

Lazybot-webapp communicates with :

- The web interface, with Socket in JSON format ;
- lazybot-master, with Socket in JSON format.

events from the web interface

connectBot

Ask to master MS to execute a new bot.

Format: JSON

Parameters:

- “login” Login object, login information (username and password if premium).

disconnectBot

Ask to master MS to stop bots.

Format: JSON

Parameters:

- “botUsername” String object, name of the bot to disconnect.

getUpdateBot

Ask to master MS to get an update of a bot.

Format: JSON

Parameters:

- “botUsername” String object, name of the bot to update.

getAllBotConnected

Ask to master MS all the nicknames of the bots connected.

Format: JSON

Parameters:

getMissionCounts

Ask to master MS to ask to the mission MS all the mission counts.

Format: JSON

Parameters:

loadMap

Ask to master to load a map around all the bots.

Format: JSON

Parameters:

- “ray” Integer object, ray of the blocs to load.

sendMessage

Ask to master MS to order bots to send a message

Format: JSON

Parameters:

- “orderBotMessage” OrderBot<String> object, message to send with the bots whom have to send it.

goToPos

Ask to master MS to order bots to go to a position.

Format: JSON

Parameters:

- “orderPositionJSON” String object. Position to go in JsonFormat.

exchange

Ask to master MS to order bots to do an exchange mission.

Format: JSON

Parameters:

- “jsonExchange” String object, exchange information.

events from lazybot-master

allBotConnected

Give all the names of the bots connected.

Format: JSON

Parameters:

- “botUsernames” String object, array of strings with all the names of the bots.

updateTotalMissionDone

Update the number of missions done.

Format: JSON

Parameters:

- “countJson” String object, mission done count.

updateTotalMissionFail

Update the number of missions done.

Format: JSON

Parameters:

- “countJson” String object, mission fail count.

updateTotalMissionRunning

Update the number of missions running.

Format: JSON

Parameters:

- “countJson” String object, mission running count.

updateBot

Update a bot (life, food, inventory, etc...).

Format: JSON

Parameters:

- “jsonBot” String object, bot updated.

4.2 lazybot-master

events from all the MS

error

Not yet implemented! Signal the error of the MS and the bot.

Format: JSON

Parameters:

- “error” String object, error message.

events from lazybot-webapp

sendMessage

Transmit a message that will be sent by a robot.

Format: JSON

Parameters:

- “orderMessageJson” String object, order in JSON with the message to send.

goToPos

Order bots to go to a position.

Format: JSON

Parameters:

- “orderPositionJson” String object, order in JSON with the position to go.

loadMap

Unused since the Map MS has been abandoned.

Format: JSON

Parameters:

- “ray” Integer object, ray of the map to load.

exchange

Send to the MS mission the exchange mission to treat it.

Format: JSON

Parameters:

- “jsonExchange” String object, exchange in JSON.

getUpdateBot

Get the information of a bot.

Format: JSON

Parameters:

- “botUsername” String object, username of the bot to get the update.

getAllBotConnected

Return to the webapp all the username of the bots actually connected.

Format: JSON

Parameters:

getMissionCounts

Ask to the MS Mission the number of mission running.

Format: JSON

Parameters:

executeMission

Inform the MS Mission that a mission has to be done.

Format: JSON

Parameters:

- “orderMission” String object, the order done.

connectBot

Execute command to connect a new Mineflayer robot.

Format: JSON

Parameters:

- “loginJson” String object, Login JSON.

disconnectBot

Disconnect bots.

Format: JSON

Parameters:

- “orderJson” String object, order Json to disconnect bots.

events from lazybot-mission

look

Ask a robot to look somewhere.

Format: JSON

Parameters:

- “orderLookJson” String object, position to look JSON.

drop

Ask robot to drop an item (ONLY ONE ACTUALLY, otherwise the robot would crash).

Format: JSON

Parameters:

- “orderDropJson” String object, order JSON to drop item.

missionStatus

Give to a robot his mission status (the actual ID of the mission he is running).

Format: JSON

Parameters:

- “orderJson” String object, mission status.

updateTotalMissionDone

Update the number of missions done (to show in the webapp)

Format: JSON

Parameters:

- “countJson” String object, count of the missions done.

updateTotalMissionFail

Update the number of missions fail (to show in the webapp).

Format: JSON

Parameters:

- “countJson” String object, count of the missions fail.

updateTotalMissionRunning

Update the number of missions running (to show in the webapp).

Format: JSON

Parameters:

- “countJson” String object, count of the missions running.

events from robots

registerBot

A robot is connecting to this MS Master. It has to be add to MasterSocket.bots which is the list of the robot actually connected.

Format: JSON

Parameters:

- “botUsername” String object, username of the bot to register.

unregisterBot

A robot is disconnecting to this MS Master. It has to be remove to MasterSocket.bots which is the list of the robot actually connected.

Format: JSON

Parameters:

- “botUsername” String object, username of the bot to unregister.

returnLoadMap

Unused since the Map MS has been abandoned. Show the map represented by a list of integer.

Format: JSON

Parameters:

- “map” List<Integer>, map represented by a list of integer.

udpateBot

Update the bot object in MasterSocket.bots.

Format: JSON

Parameters:

- “jsonBot” String object, Bot object in JSON.

missionDone

Inform the MS Mission that a mission has been done successfully.

Format: JSON

Parameters:

- “jsonMissionId” String object, id of the mission.

missionFail

Inform the MS Mission that a mission failed.

Format: JSON

Parameters:

- “jsonMissionId” String object, id of the mission.

4.3 lazybot-mission

events from lazybot-master

getMissionCounts

Return all the counts of the missions.

Format: JSON

Parameters:

missionDone

Event receved when a step of a mission is ended. If there is a next step (mission can continue) do it, else if the mission is finished (don't have next step) remove the mission from the list of missions running.

Format: JSON

Parameters:

- “jsonMissionId” String object, id of the mission.

missionFail

Event receved when a step of a mission fail. Delete the mission from the list of missions running, increment the counter “totalMissionFail” and send the new counters to the master.

Format: JSON

Parameters:

- “jsonMissionId” String object, id of the mission.

exchange

Execute the exchange mission.

Format: JSON

Parameters:

- “missionJson” String object, ExchangeMission object in Json.

4.4 lazybot-map

This microservices is not implemented.