# DEVELOPMENT OF A LORA BASED MULTIPLAYER GAME USING AN ARDUINO APPLICATION

## INTERMEDIATE REPORT

COURSE: WIRELESS MOBILE NETWORKS
FACULTY: ANTÓNIO GRILO

MIGUEL RODRIGUES Nº76176
HENRIQUE PIEDADE Nº75546
GROUP 19

15/04/2018

# 1 Introduction

In this project, we aim to develop a multiplayer shooting game, where an Arduino is used as our weapon, communicating through Semtech's LoRa Technology, instead of regular WiFi communications. LoRa is an innovative long range and low power wireless platform that implements the LoRaWAN protocol. It is through this protocol that communications are made, allowing for secure and reliable IoT networks.

In this intermediate report we aim to expose how our game will be made, i.e, the hardware needed, the technologies used, the structure and functionalities of both and how we will test it to ensure it is working correctly.

# 2 Hardware

## 2.1 Arduino Uno

The Geekcreit Arduino Uno board will be used as the gun itself, providing support for the remaining hardware. In it will be placed a LoRa Shield, with a LoRa Bee and a GPS module and it ill have connected two digital push buttons and a 5V buzzer. Its inbuilt LEDs will also be used to signalize different actions.

Figure 1: Arduino Uno

## 2.2 LoRa GPS Shield

In order to communicate using the LoRa protocol, it is used a Dragino LoRa GPS Shield for Arduino, which include a LoRa Bee for LoRa communications and a Quectel L80-R GPS module.
The LoRa Bee is in charge of sending and receiving data, so that actions can be performed to play the game. The GPS module will provide the user's location and orientation, which will both be fundamental for the game to be played.

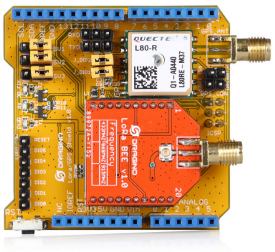

Figure 2: LoRa Shield

## 2.3 Digital Push Buttons

The two DFRobot Digital Push Buttons (SKU:DFR0029) will be used to deploy mines and shoot other players. The green one will be used for mines and the red one for shooting.
With their inbuilt resistor, there's no need for external resistors between the buttons and the Arduino board.

Figure 3: Digital push button

### 2.4 Buzzer 5V

The PTR000607 model 5V Buzzer from PTROBOTICS, that operates around the 2kHz range, will be used to signal when a player has been hit by another player or when he has stepped on a mine deployed by another player, indicating its death. To differentiate between the two types of deaths, different tones, either in duration or frequency,will be used. As we don't have the buzzer yet, it has not yet been decided how the tones will differ.
With its inbuilt resistor, there's no need for external resistors between the buzzer and the Arduino board.


Figure 4: 5V Buzzer

## 3 Arduino Flowchart

In this section, we provide an overview of what happens with the usage of the Arduino board for the purpose of this game.

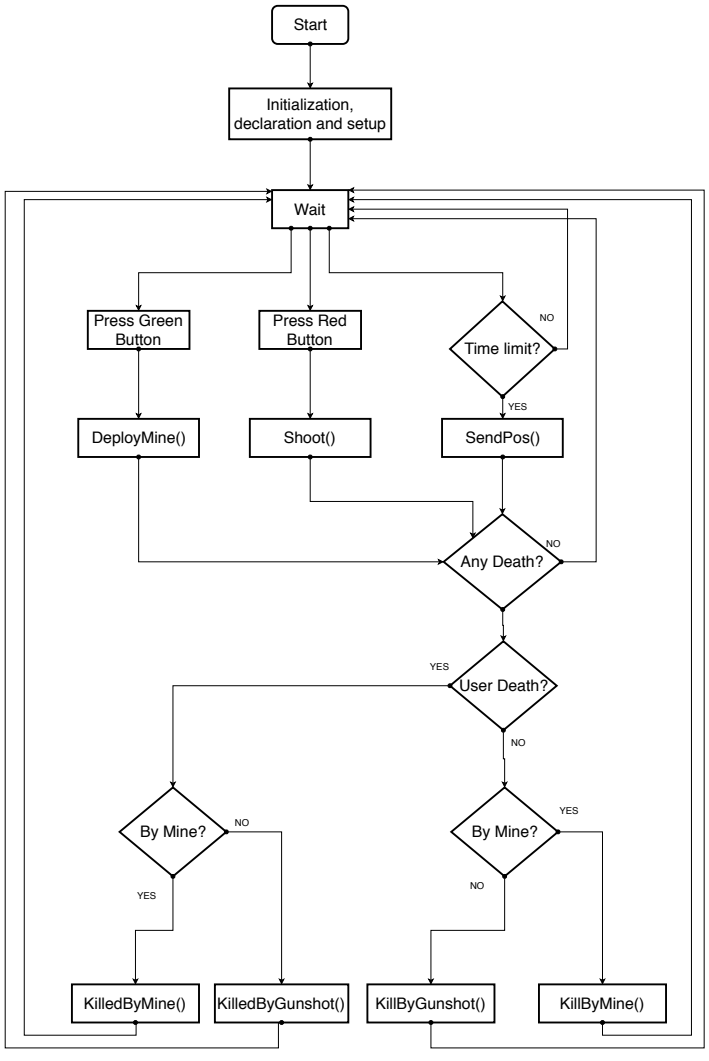


Figure 5: Arduino Flowchart

In Figure 5 we show the different actions a player can take (either pressing one button or the other), which will send a message to the Web server, creating a new mine (DeployMine()) or trying to hit another player (Shoot()). Besides that, the Arduino program will also ensure that the position of the player is updated regularly with a certain time interval between updates (SendPos()), which will also be updated when the user shoots or deploys a mine. As the LoRa protocol needs an uplink message to send downlink information (discussed in the LoRa section ), the updates regarding deaths will only be sent when the position update is made or when any of the buttons is pressed, as in all cases there's an uplink message.

If the player was killed, it will trigger the buzzer, according to type of death he suffers (KilledByMine() and KilledByGunshot()). If the player killed someone, a combination of LEDs will be used to show it, with different combinations for a gunshot (KillByGunshot) or a mine (KillByMine()). In case more than one action is required to be sent, the most recent ones will be sent (until the packet is full) and the buzzer/LEDs will trigger sequentially for each action.

# 4 LoRa

In order to communicate between the Arduino and the Web server with LoRa, we must use the The Things Network platform, an IoT network, which uses LoRaWAN, with several gateways through which we can connect to the Internet. To begin using it, we msut register our Arduino on the network, using its EUI.

As this network provides several integration options with the applications registered within, like a Database integration, we communicate with our Web server using the HTTP Integration, allowing us to send uplink messages and receive downlink messages over HTTP, in this case using the POST method, to a link specified in the The Things Network integration page.

The messages sent over the payload will be different according to what suits the player action's needs. To update the position or deploy a mine, the ID and GPS coordinates will suffice. To shoot, we'll have to include the player's orientation. There are limitations regarding the amount of uplink and downlink messages sent, hence the time limit between position updates, instead of updating constantly.

As for the downlink messages, different parameters will be defined so that the Arduino can recognize how to use the LEDs and the buzzer corresponding to what happened (e.g. death by mine or by gunshot).

# 5 Web Server

The needed Web server will be implemented using PHP. It will be hosted on Técnico's available cluster, sigma, where there is already an available hosting service.

The purpose of the server will be to receive messages from the Arduino board and interact with the database, in order to register players, movements and actions related with the game.

There will also be a few features included in the website as the score of each player, the last know positions and a small history of actions taken (e.g: last 10 deaths).

# 6 Databases

As for out database, it will also be stored at the aforementioned cluster, using MySQL and it will consist on the storage of three types of objects, as described on Figure 6:

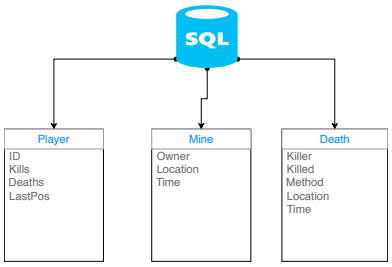


Figure 6: Database

## 6.1 Player

The Player structure will have information on the player's ID, the number of kills and deaths he has and the last known position which will be updated over the time.

## 6.2 Mine

The Mine structure will have a field for its owner (i.e., the player who left that mine), the location where it was set and the time, so that it doesn't stay there forever, being eliminated after a certain period of time.

## 6.3 Death

The third structure, Death, will have information on all the deaths that occur, serving as a history. It records who killed who, how, where, and when.

# 7 Web Server and Database Interaction

Using PHP, our web server will interact with the data stored for two different purposes: extract info and create or modify data. For each action a player takes, the database has to be used and that is made after the Web server receives the message from the Arduino containing what to do. In order to refresh a players current positions, each time the player sends its coordinates, the LastPos field of the structure with the corresponding ID must be actualized, as well as when he shoots his weapon or plants a mine. When a mine is planted, it must also be created a new Mine structure, with the owner ID, the location of the mine and the time it was set up.

When a player shoots someone, the player database must be searched for a certain range of coordinates that would be affected by that bullet and return that information to both the killer and the victim. In similar

fashion, whenever a player updates its position, the Mine database must be searched for a certain range where the mine explosion would kill that player.

As a complement to the game, both the Player and Death databases must be searched regularly in order to display that information on the server, as mentioned before.

In order to ensure a good practice, the database connection will be opened and closed as needed with every database querry.

# 8 Testing

As the only real player of this game is our board, in order to test all of the functionalities, we will insert custom data into to the database to trigger the different events and ensure their correct behaviour. These tests will include:

- Adding new players
- Adding and removing death history (to display in the Web server)
- Deploying a mine (both DB entry and using Arduino)
- Stepping over a mine (both DB entry and using Arduino position)
- Shooting with Arduino wihout killing
- Shooting with Arduino killing someone
- Arduino getting shot at
- More than one update to be made for a user

# 9 Final Remarks

The project itself is very interesting as it gives us the opportunity to learn and explore some technologies with which we aren't acquainted with, giving us an opportunity to test a real-time interaction between (what is supposed to be) a set of agents (players) in an architecture we developed.

Furthermore, it gives us the opportunity to work with LoRa and understand some of its mechanisms so that we can enter the world of IoT and explore the current paradigm of wireless communications between different technologies.