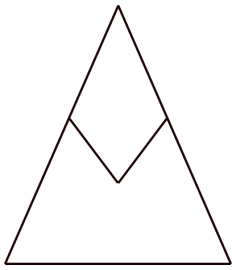
Bern University of Applied Sciences

Project 2

Alpinist Tracker & Alerting System
ATAS



Author:
Martin Schmidli

Teacher:
Mohamed Mokdad

Bern, June 25, 2017

Contents

Chapter 1

Project description

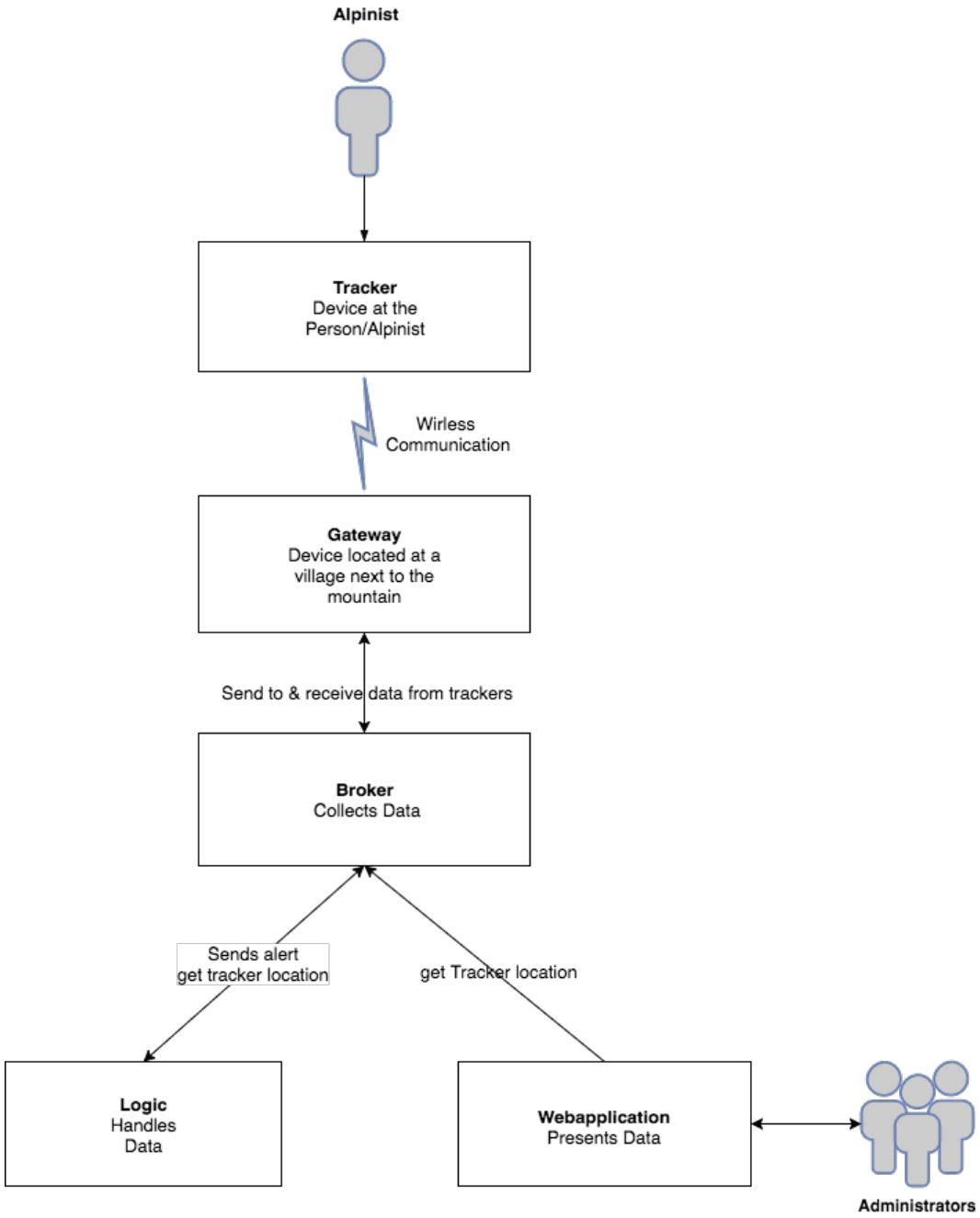
1.1 Introduction

The Alpinist Tracker and Alerting system short ATAS has two main functionalities.

1. Track peobles location which are located on a mountain for example skiers, hikers... short called alpinists.
2. Alarm a rescue team if the alpinist had an accident

1.2 Systemarchitecture, Abstract

This section will provide an abstract overview which kind of components are used in the ATAS System.
The sytem will be devided into different modules. All the modules and users of the system are explained on the next page.



1.2.1 System Users

Alpinist

Generalization: Skier, hiker or any other person in the mountains.

Administrators

The administrators of the system are peoble working for a tourism region, hospital or rescue teams like the Rega.

1.2.2 System Modules

Tracker

Imagine a small mobile device, following called **tracker**. The Tracker can be carried by an **alpinists** for example in a bagpack.

The tracker device is equipped with an **alarm-button** and a **speaker**. With the alarm-button, the alpinist can inform the administrators about an accident. For example if the alpinist broke its knee.

The speaker is used to inform the alpinist, with an alarming sound, that he's situated inside a dangerzone and he would better leave this zone immediately. A dangerzone is a place with a high possibility for an accident, for example, an area with a high possibility for an avalanche.

The tracker is sending its GPS position to a receiver station, following called **gateway**.

Gateway

A device which is able to talk to multible trackers. The gateway can be installed in a village/ building in the valley next to the mountains.

All the gateway are sending the received locations from the trackers to a central datamanager called **broker**.

Broker

The broker collects data and stores them. The Broker provides an Interface where the webapplication can grab all the data to present them.

Webapplication

The administrators can use a **webapplication** / webpage to

- monitor the actual position of the persons on the mountains.
- define dangerzones

Logic

Software / service which calculates if a tracker is inside a dangerzone. If the tracker is inside a dangerzone it will send an alarm to the tracker. The alarmsignal is relayed over the Broker.

1.3 Usecase

So why is the ATAS system useful? The ATAS system can be used for:

- If an avalanche is triggered in the mountains, its position can be compared to the position of the trackers. If at the time of the incident, a tracker was in this area, the rescue teams can be sent to do a rescue operation. This immediate action will give the victims important seconds and may save lifes.
- If peoble were in the avalanche and the tracker survived, the device could keep sending data and support the rescue team in their mission bye pointing them to the location of the persons. If no transmission is possible through snow, the rescue team has at least its last GPS location, this might reduce the amount of time to find the person. ATAS is not planned as a replacement for avalanche transceivers, it should be seen as a supplement.
- If a tracker is moving towards a dangerzone, for example an area with rockfall, the user can be informed with the tracking device.
- If an alpinist had an accident in the mountains, the person can send a rescue signal with the tracker device by triggering the alarm-button.

1.4 Technologies

At first glance this hole system must look very laborious to a user. Why should a person have to take another device with him on his hikes? He could easily use his smartphone and an App to send his location. This statement is true if we consider only the touristic ski areas of switzerland. In most ski areas the phone reception is wonderful. If we leave those ”safe” areas and go to a higher altitude, the reception gets worse or disappears completely.

This is the reason why for this project other technologies had to be found.

For this project the following technologies were used:
MQTT, LoRA, LoRaWAN

You should get yourself familiar with these technolgies to fully understand the following chapters.

1.5 Demarcation

This section defines which kind of tasks are to do during this project. These goals have been set by the project leader.

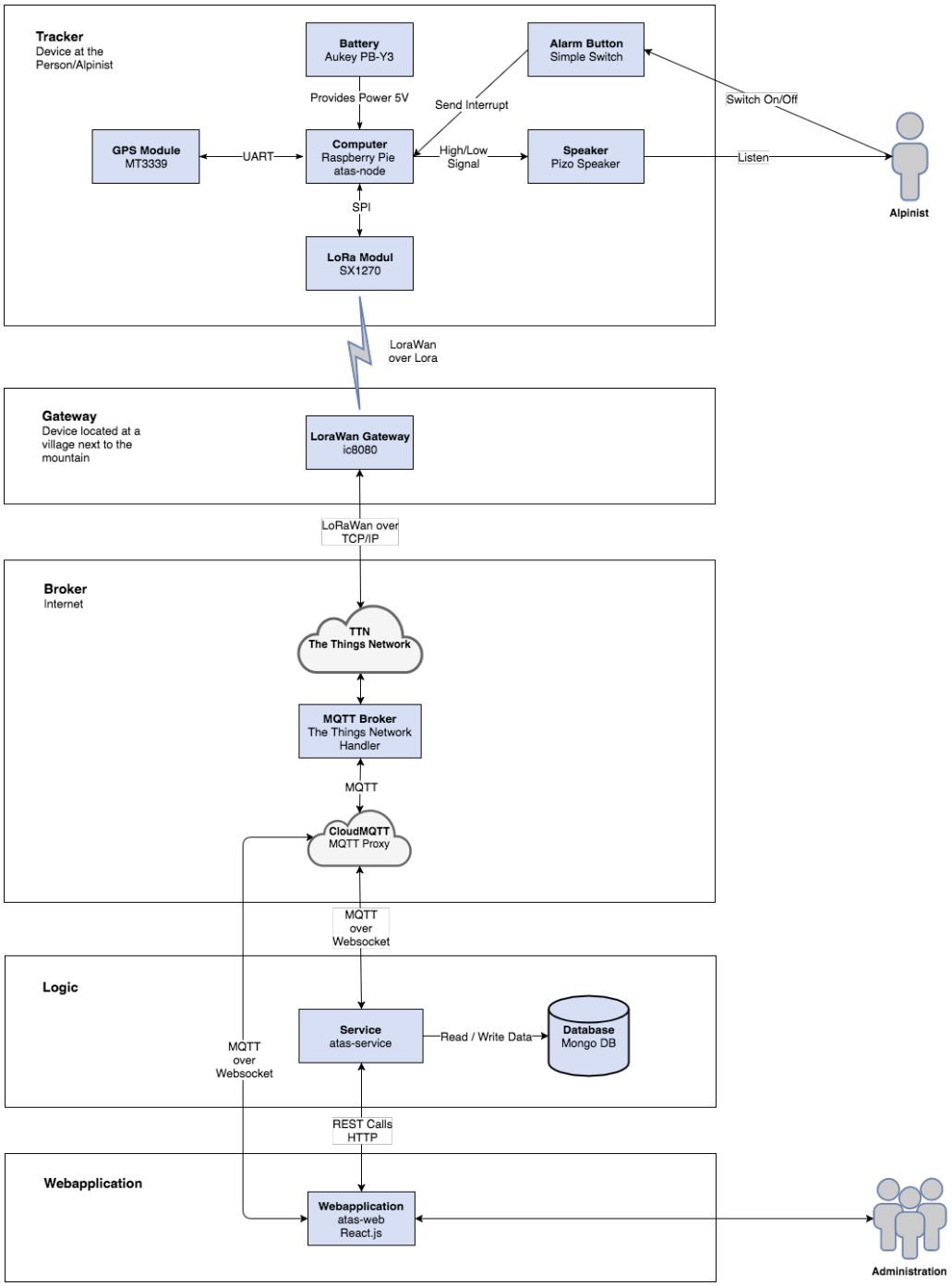
- Build a tracking device
  - The alpinist should be able to trigger a manual alarm and inform the rescue team of an accident
  - the tracker should track the position of the user and send the data to a data collector (broker)
- Build a webpage, which enables the administrators to
  - view the location of the trackers
  - manage dangerzones
- Build a software / service which controls the position of the trackers. If a tracker dwells inside a dangerzone the tracker has to be alerted.
- Connect all the modules
- Setup a broker system

Chapter 2

Systemarchitecture

This chapter will provide a detailed overview how the ATAS System was implemented. It will be more technical and show in detail how the modules of ATAS interact with each other.

2.1 System Overview



2.2 Tracker

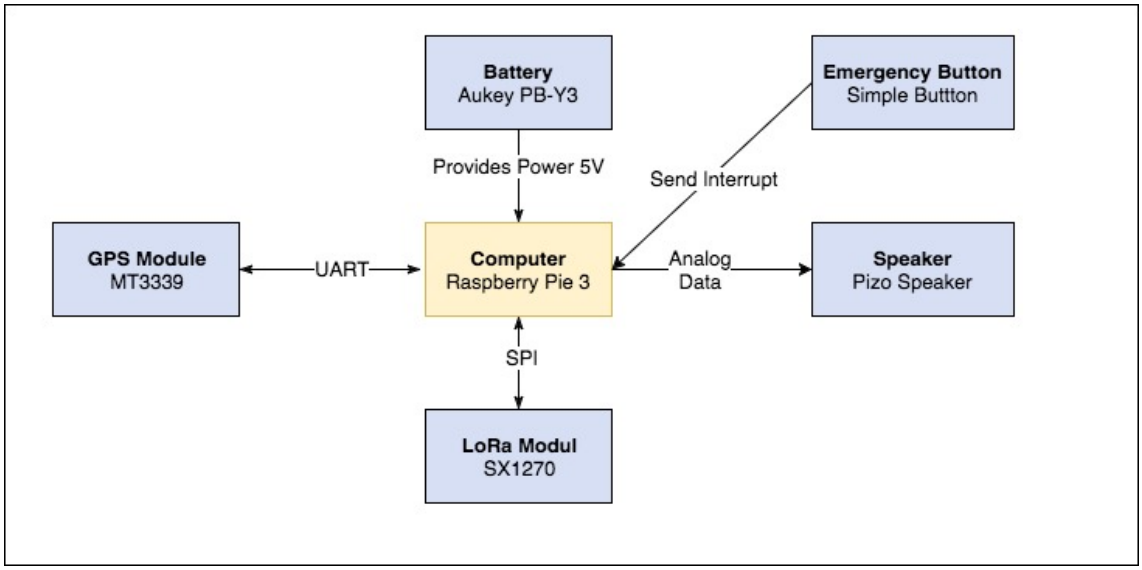
The tracker itself contains different components.

2.2.1 Computer

As my knowledge for hardware close porgramming (c, c++ ...) is failry limited, I decided to use a hardware plattform, which handles most of the system tasks for me. The computer had to be small and doesn't use to much energy. I decided to use a Raspberry Pie 3 single board computer.

System Overview

The Raspberry Pie is connected to other devices



Tasks

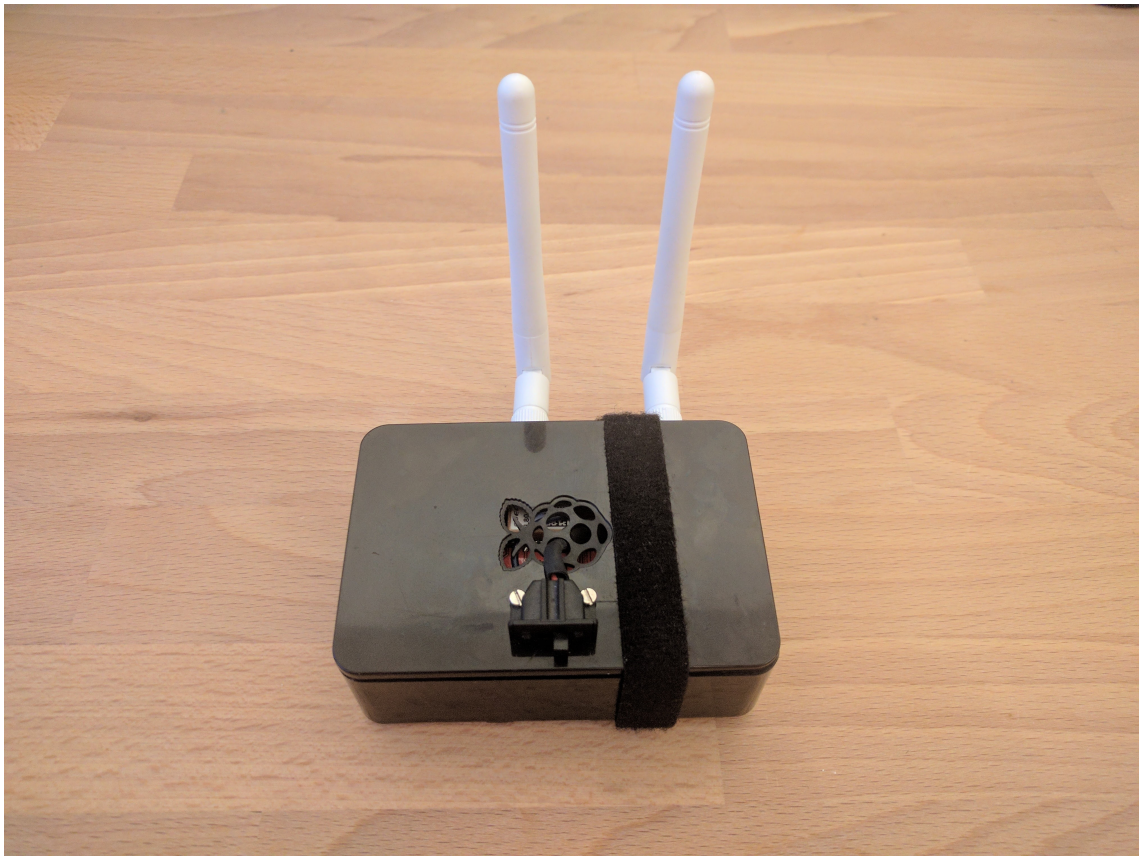
The Raspberry Pie computer is used to

- Recieve GPS data from the GPS module via UART
- Recieve and send data to the Lora Module via SPI
- Listen for signals from the alert-button
- Enables or disables the speaker

Finished product

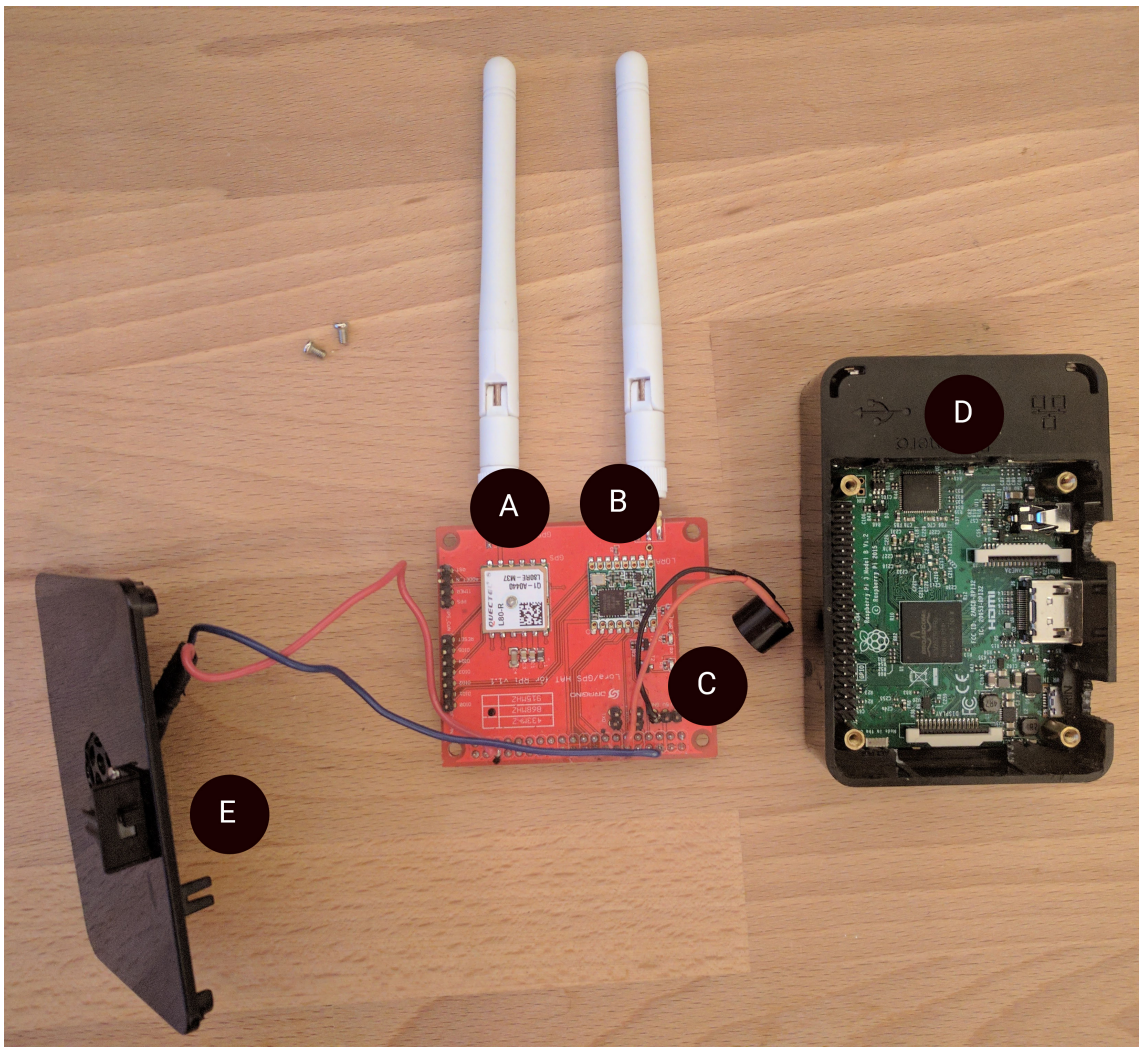
The following pictures show the final tracker I built during this project

Picture of the assembled tracker



The alarm-button was placed on top of the case. All the other components are safely stored inside a case.

Picture of the disassembled tracker



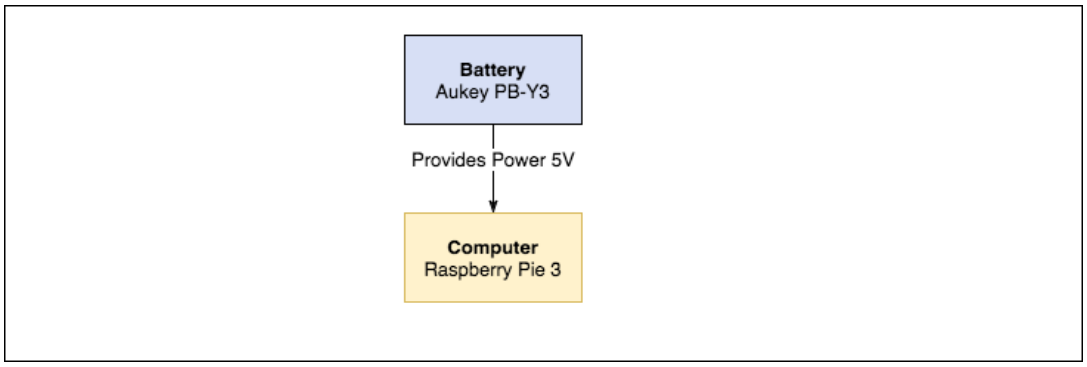
- A: GPS Module with the corrsponding antenna
- B: LoRa Module with the corresponding antenna
- C: Piezo Speaker
- D: Raspberry Pie computer
- E: Alarm-button

2.2.2 Battery

The tracker needs to mobile so we can put it in our backpack. To power the tracker I attached a battery pack over USB.

Type: Aukey PB-Y3 Battery Pack
Power: 30000mAh
Physical Interface: Attached to the Raspberry Pie over Micro USB.

Diagramm

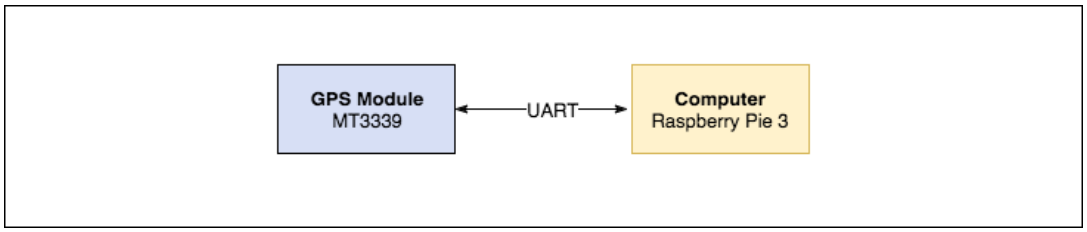


2.2.3 GPS Module

The GPS Module provides the GPS location to the tracker.

Type: MT3339
Interface: Acessable via UART.

Diagramm



Tasks

The GPS module will be used to

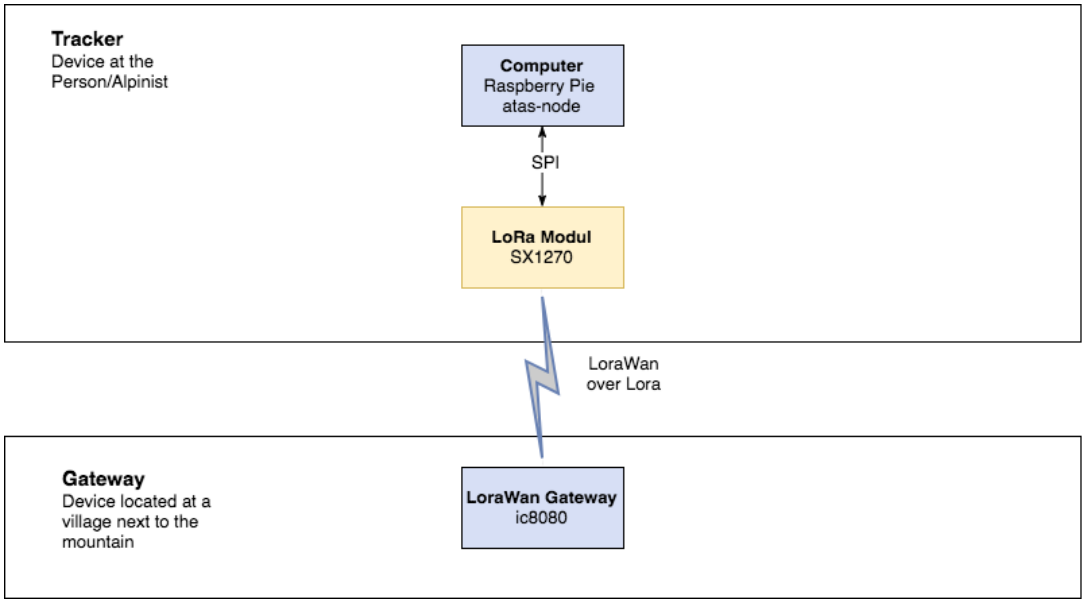
- Get the GPS Location(Longitude, Latitude) of the Tracker
- Sends GPS data to the Raspberry Pie via UART

2.2.4 LoraWan Modul

The LoRaWan Modul establishes the wireless connection to a LoRaWan gateway.

Type: LoRa Module Semtech SX1276
Interface: Acessable via SPI

Diagramm



Tasks

The Lora module is used to

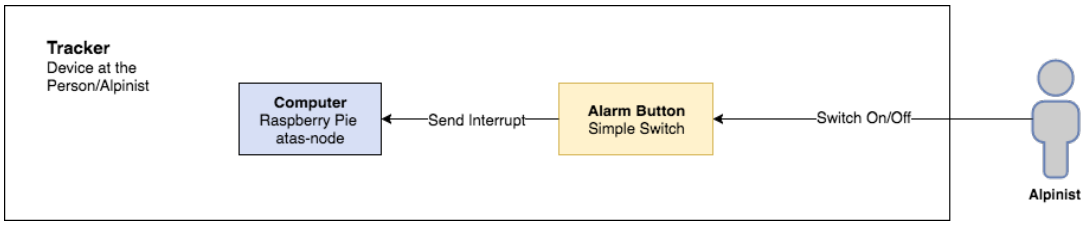
- Receive and transmit data to LoRaWan gateway.
- Receive and transmit data to the Raspberry Pie over SPI.

2.2.5 Alarm Button

A simple hardware switch. Can be put to On or Off.

- If the switch has been triggered by the alpinist, the GPIO input on the Raspberry Pie is set to Low (On / Alert) or High (Off / no Alert)
- This simple switch gives the alpinist the possibility to trigger an alarm manually

Diagramm



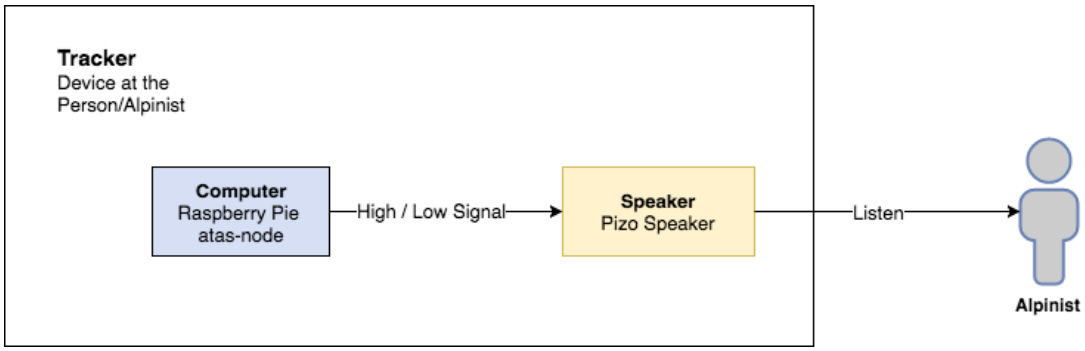
2.2.6 Speaker

A simple Piezo buzzer.

Tasks

- Gets enabled by the Raspberry Pie
- Generates an alamring signal, which indicates, that the tracker is inside a dangerzone.

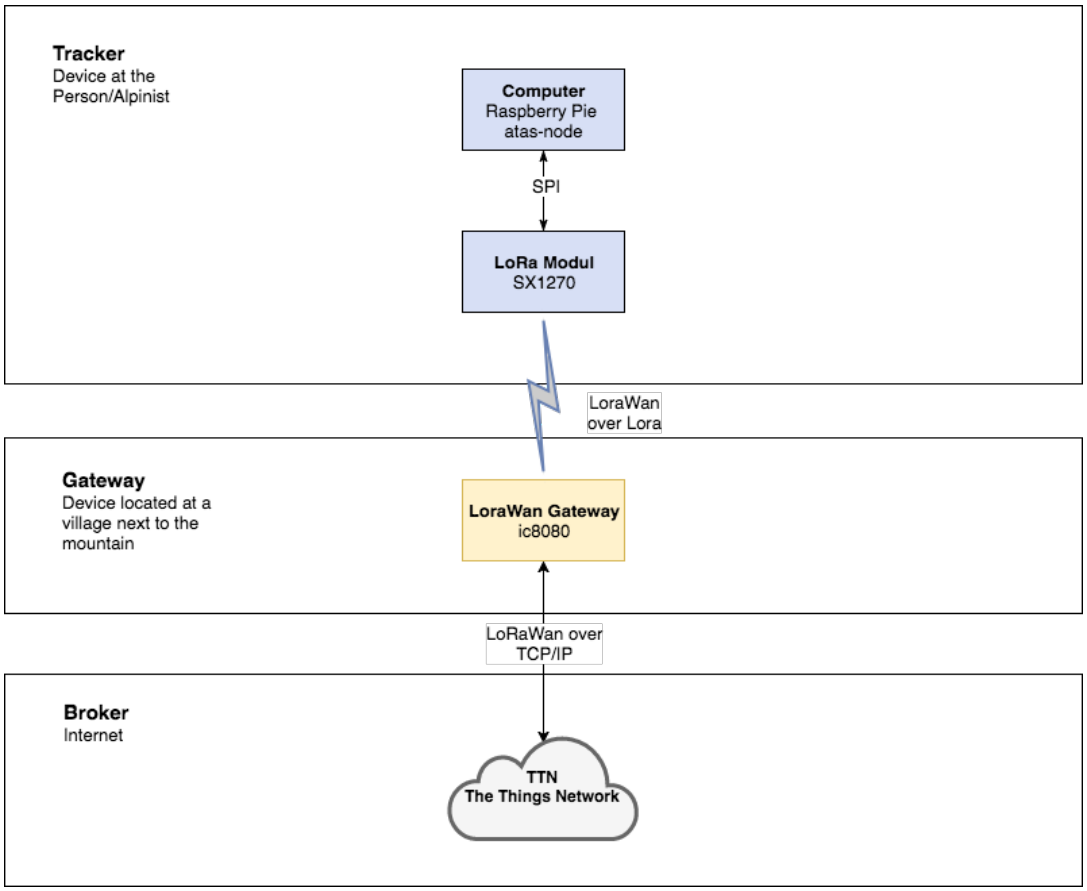
Diagramm



2.3 Gateway

The gateway relays data between the trackers and The Things Network (TTN). What kind of fucntionality TTN is providing is explained later in this docuent.

Diagramm



Tasks

The Lora Gateway will be used to

- Receive and transmit data to the Cloud Network The Things Network (TTN) over LoRaWan via TCP/IP
- Sends and receive datat to the Trackers over LoRaWan via Lora

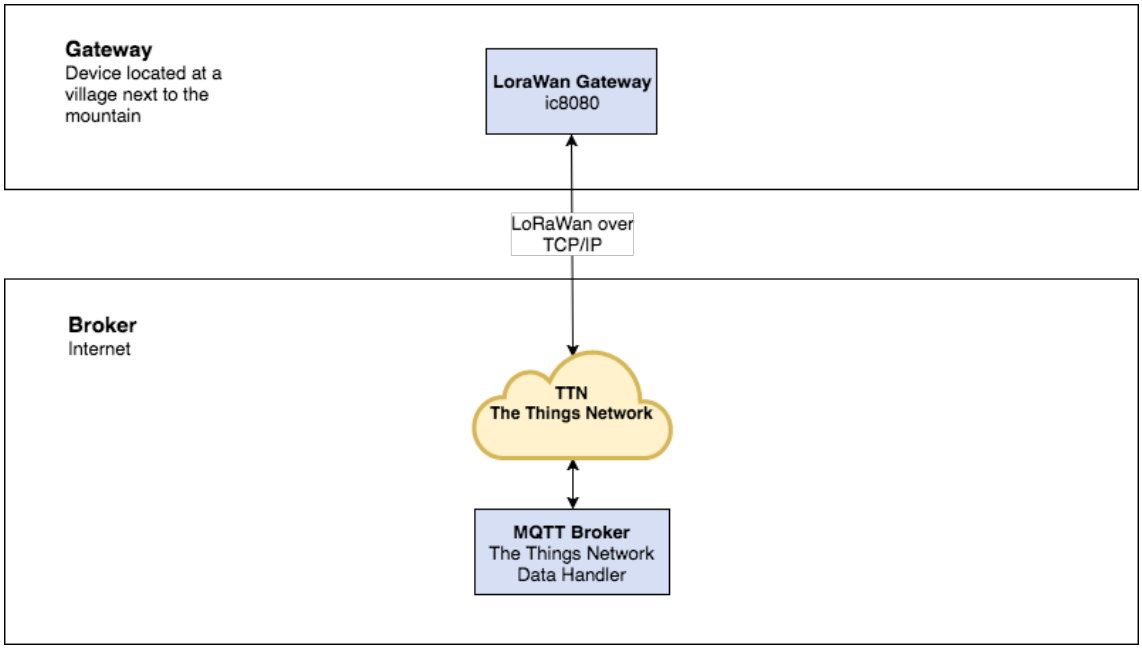
2.4 Broker

The broker has been split up in subcomponents. The broker service hasn't been developed by myself. I use two platforms the Things Network (TTN) and CloudMQTT.

2.4.1 The Things Network short TTN

The Things Network short TTN is an internet platform whichs allows us to connect LoRaWan devices. TTN will send (Downlink) and receive (Uplink) data from the trackers via gateway. TTN will collect and store all the data which the gateways send to it. TTN provies a MQTT Broker to access the data more easily.

Diagramm



Tasks

- Manage trackers
- Manage gateways
- Provides MQTT broker
- Send and Receive data from the trackers via gateway

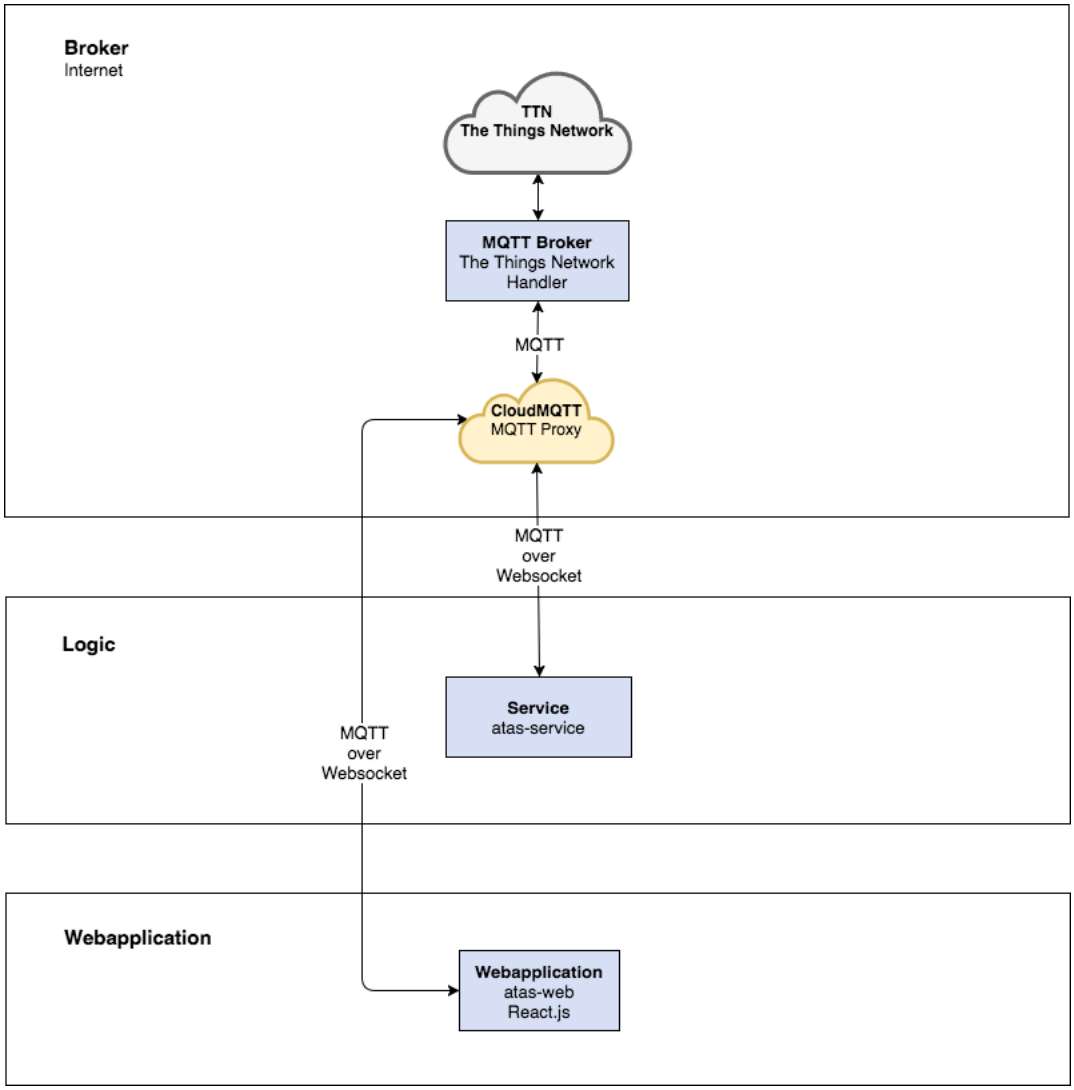
2.4.2 CloudMQTT

To this date, the TTN MQTT broker does not provide the possibility to access the data via websocket interface. To bring MQTT over websocket to a webapplication, another service was needed. I decided to setup CloudMQTT.

CloudMQTT mirrors the TTN MQTT Broker Topics and provides an websocket interface. Our webapplication and the atas-service can use this interface to request and publish data from the trackers.

All the data published to CloudMQTT are published to the TTN MQTT broker aswell. All the topics of the TTN broker are subscribed from the CloudMQTT service.

Diagramm



2.5 Software

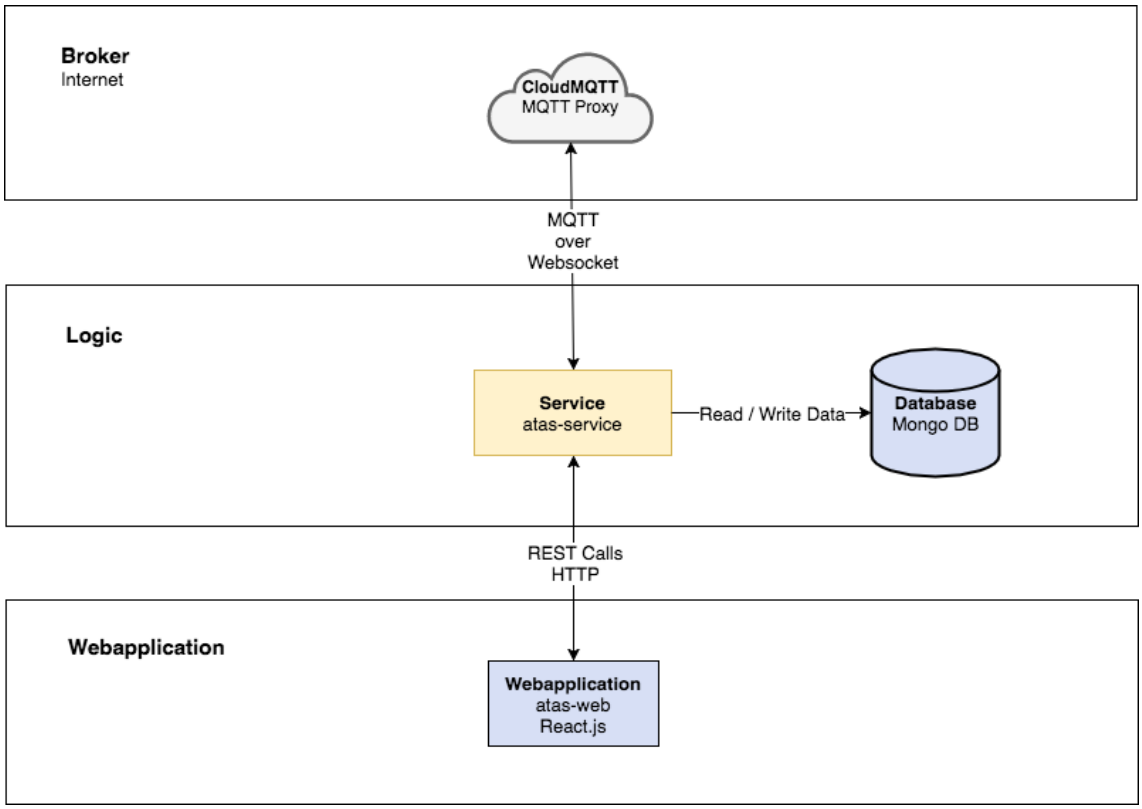
2.5.1 Atas-Service

Atas-Service calculates if a tracker is inside a dangerzone. The service is comparing the GPS location of the tracker with the geographical area of the dangerzone. If the tracker is inside a dangerzone it will send an alarm to the tracker. The alarmsignal is send to the TTN Network over MQTT via CloudMQTT.

Atas-Service provides a REST interface. Dangerzones created with the webapplication are sent to the atas-service over REST. This allows the webapplication to get and store the information about dangerzones. The database used in this project is a Mongo db.

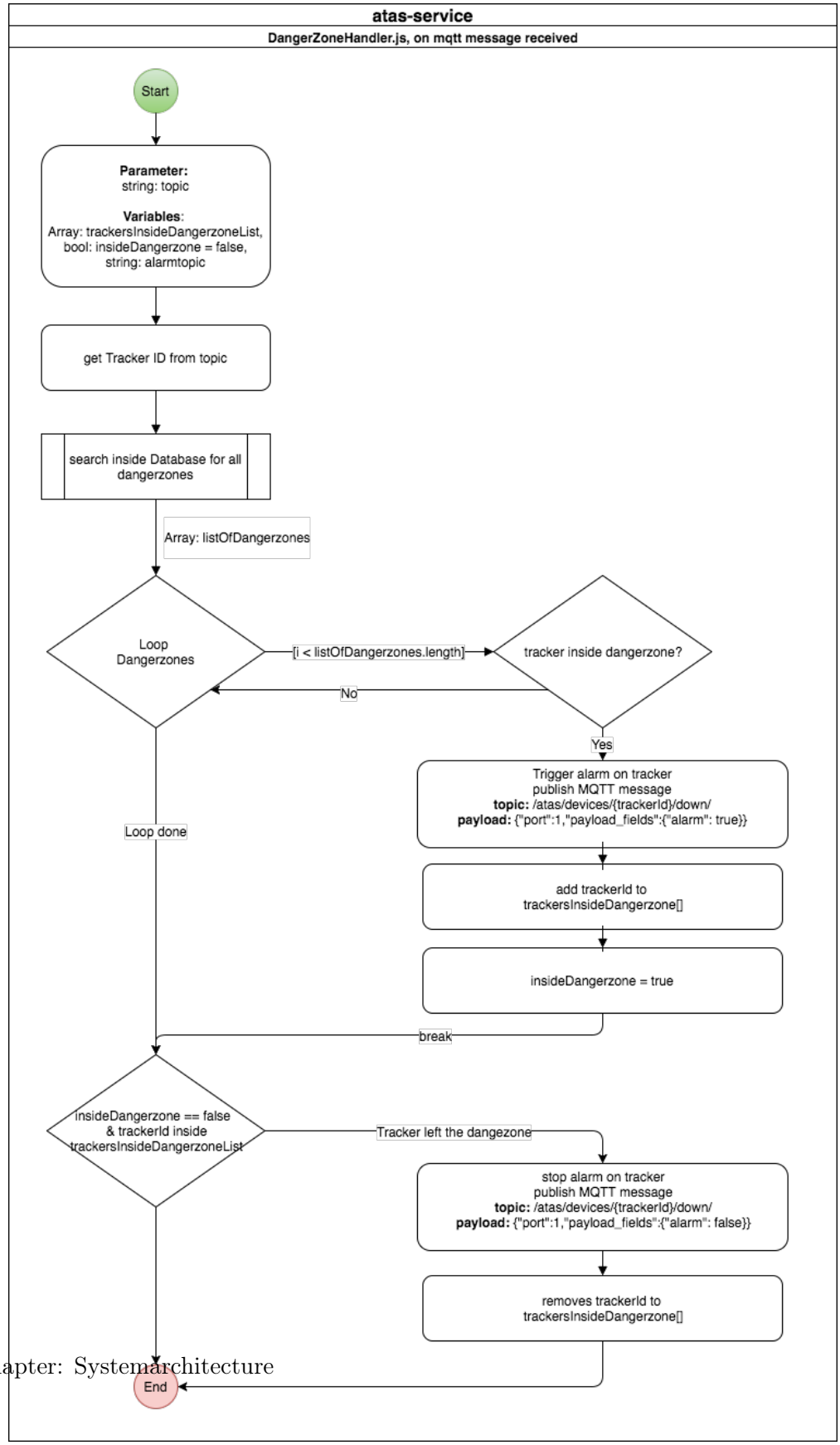
Atas-service is a node.js application written in Javascript.

Diagramm



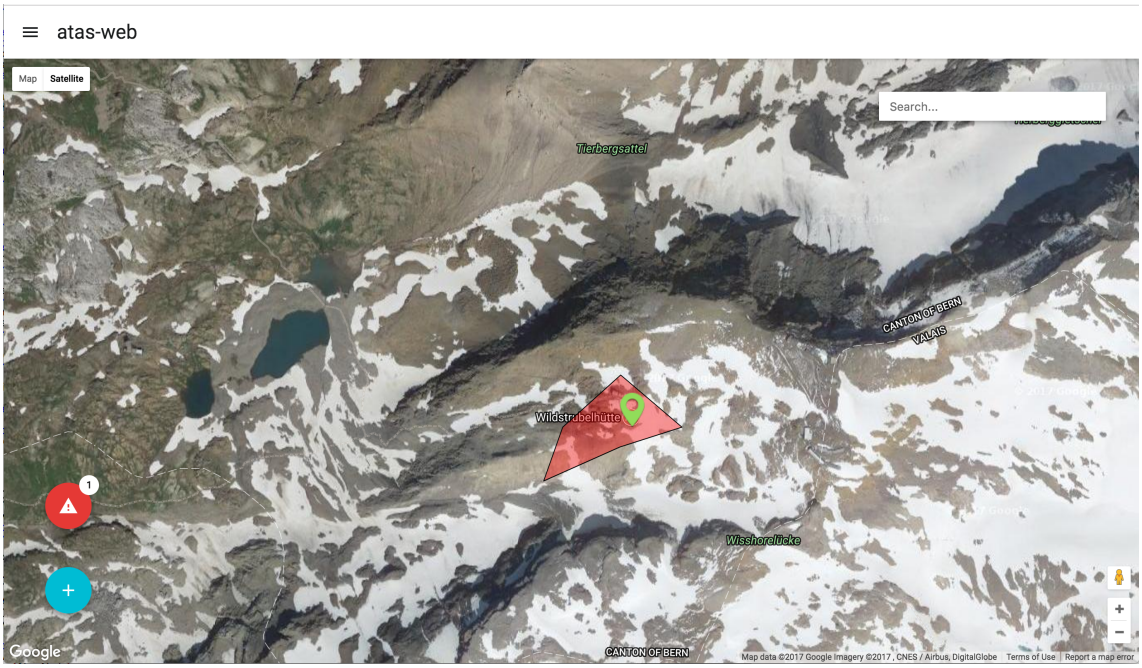
Process - MQTT Message arrives

The flow chart on the next page displays the process when a new MQTT message with GPS location data arrives. The flowchart doesn't represent the actual code. Dummy syntax /code was used to make the chart as understandable as possible.


atas-service
DangerZoneHandler.js, on mqtt message received
Start
Parameter:
string: topic
Variables:
Array: trackersInsideDangerzoneList,
bool: insideDangerzone = false,
string: alarmtopic
get Tracker ID from topic
search inside Database for all dangerzones
Array: listOfDangerzones
Loop Dangerzones
[i < listOfDangerzones.length]
tracker inside dangerzone?
No
Yes
Loop done
Trigger alarm on tracker
publish MQTT message
topic: /atas/devices/{trackerId}/down/
payload: {"port":1,"payload_fields":{"alarm": true}}
add trackerId to trackersInsideDangerzone[]
insideDangerzone = true
break
insideDangerzone == false & trackerId inside trackersInsideDangerzoneList
Tracker left the dangezone
stop alarm on tracker
publish MQTT message
topic: /atas/devices/{trackerId}/down/
payload: {"port":1,"payload_fields":{"alarm": false}}
removes trackerId to trackersInsideDangerzone[]
End

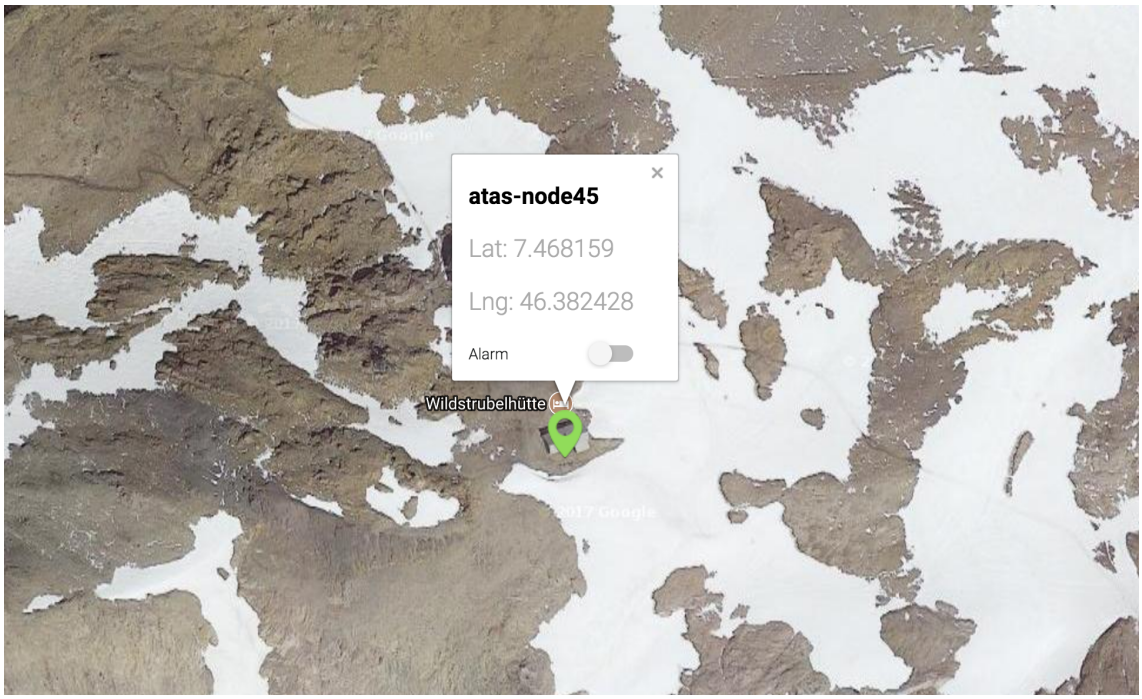
2.5.2 Atas-Web

Atas-Web is the management interface of the ATAS System. Atas-Web is a webapplication. The application is written in Javascript and leverages the React.js library.



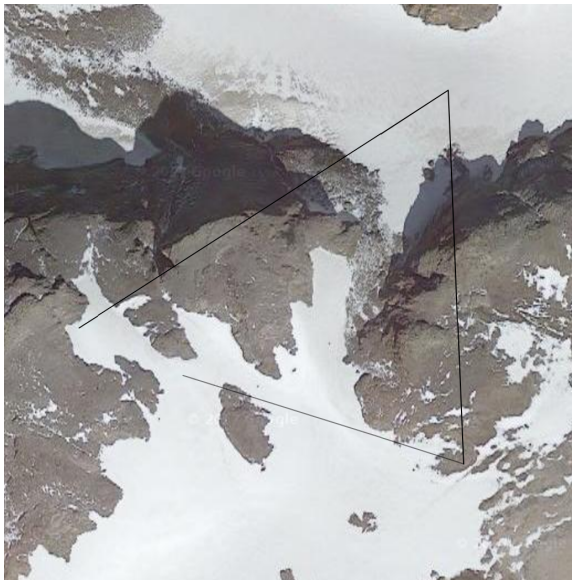
Functionality

- **Map**
  All the trackers are displayed on a world map. The trackers are displayed as a green symbol. If we click on a tracker, we see the actual GPS location.

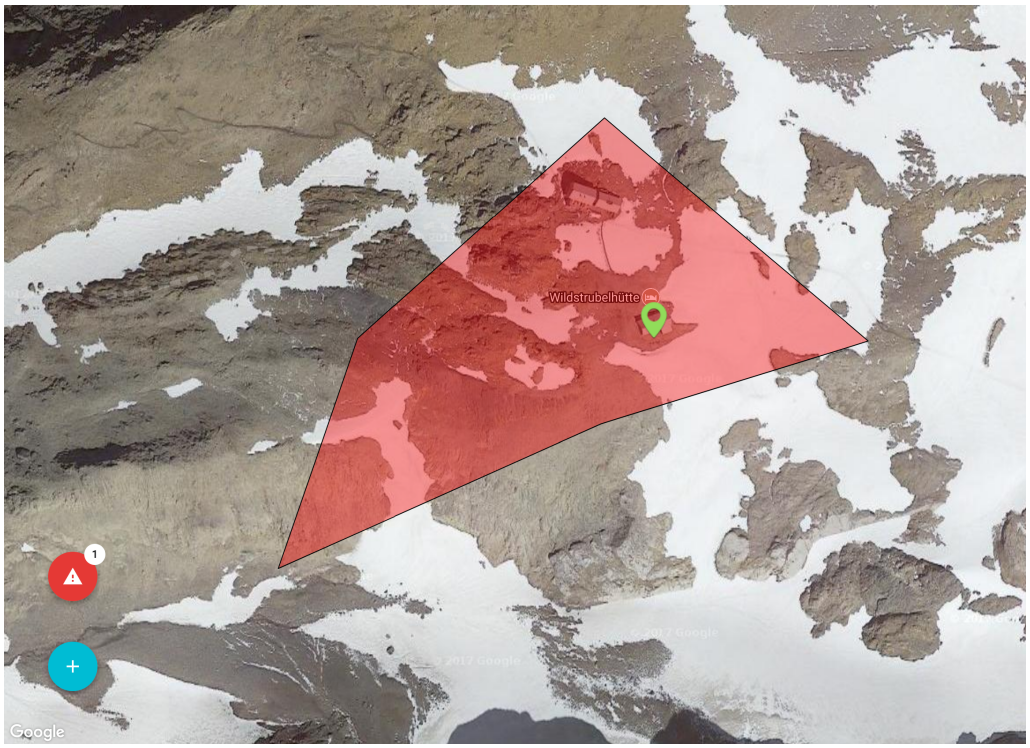


- **Dangerzone Mangement**
  We can easily create and delete dangerzones on the map. We can click on the Plus sign in the bottom left corner to change into the drawing mode

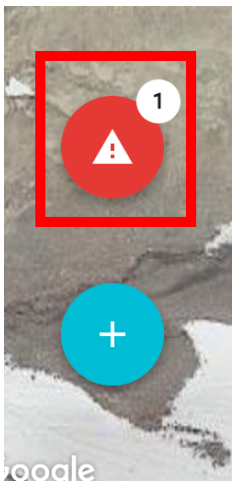
Once we switched in the drawing mode, we can draw a polygon which represents the dangerzone on the map.



An example of a drawn dangerzone. As we can see a tracker is inside the dangerzone.

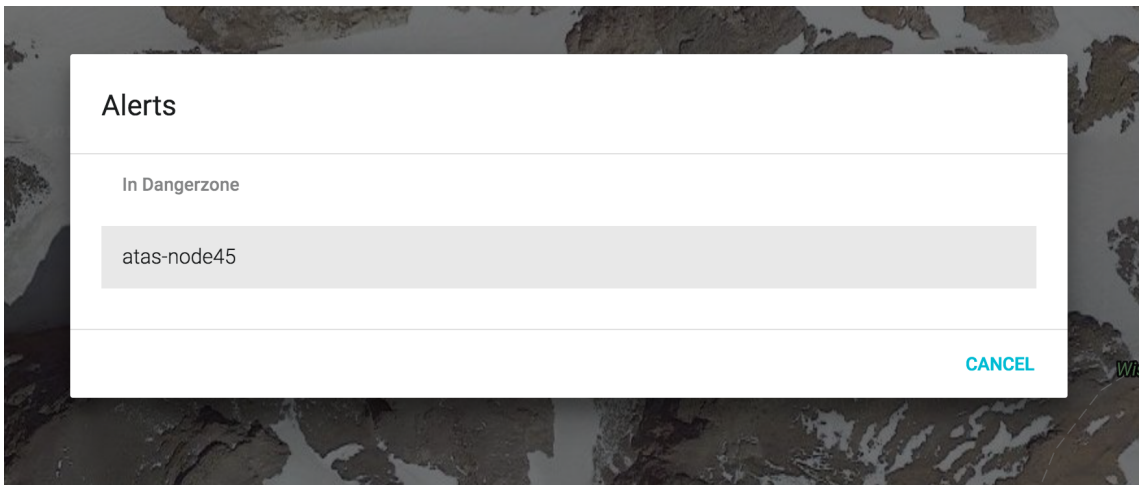


- **Alert View**
  If we click on the Alert button

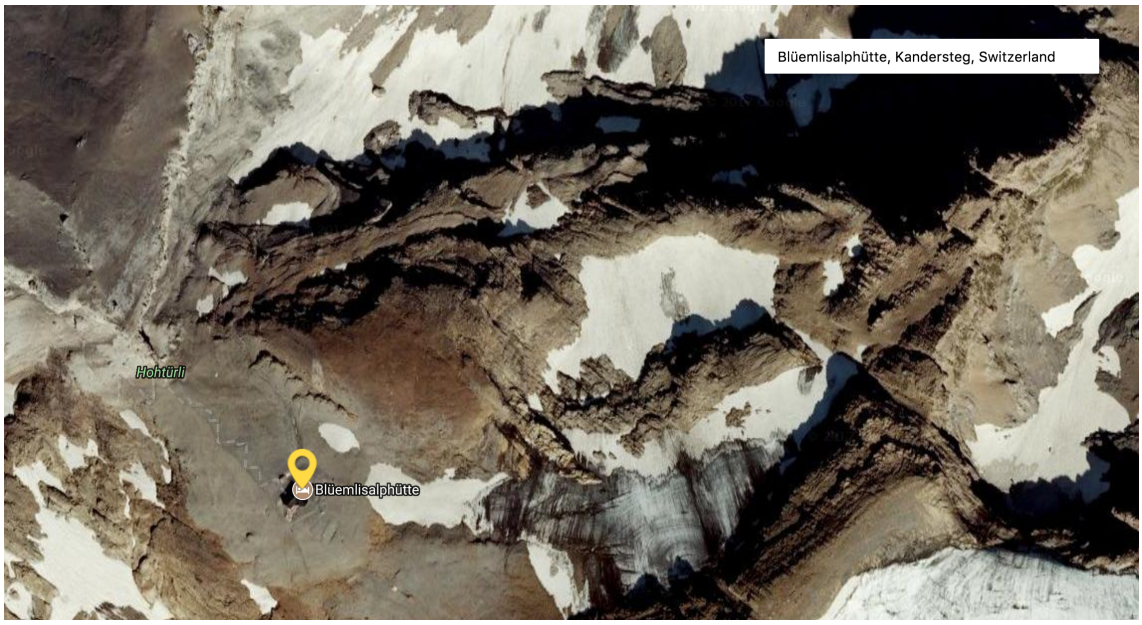


we get an overview of all the alpinists requesting a rescue team and the trackers which resides inside a dangerzone.

In this example the tracker atas-node45 resides inside a dangerzone.



- **Search**
  We can use the search bar to search for a location. Found locations are displayed as a yellow symbol.



2.5.3 Atas-Node

Atas-Node is the software component running on the tracker computer. The software is written in c++.

To make the communication between atas-node and the GPS Module as easy as possible I decided to install GPSD. GPSD is a daemon running on the linux Operating System on the tracker. GPSD provides us a simple interface and a c++ library to query the actual GPS position.

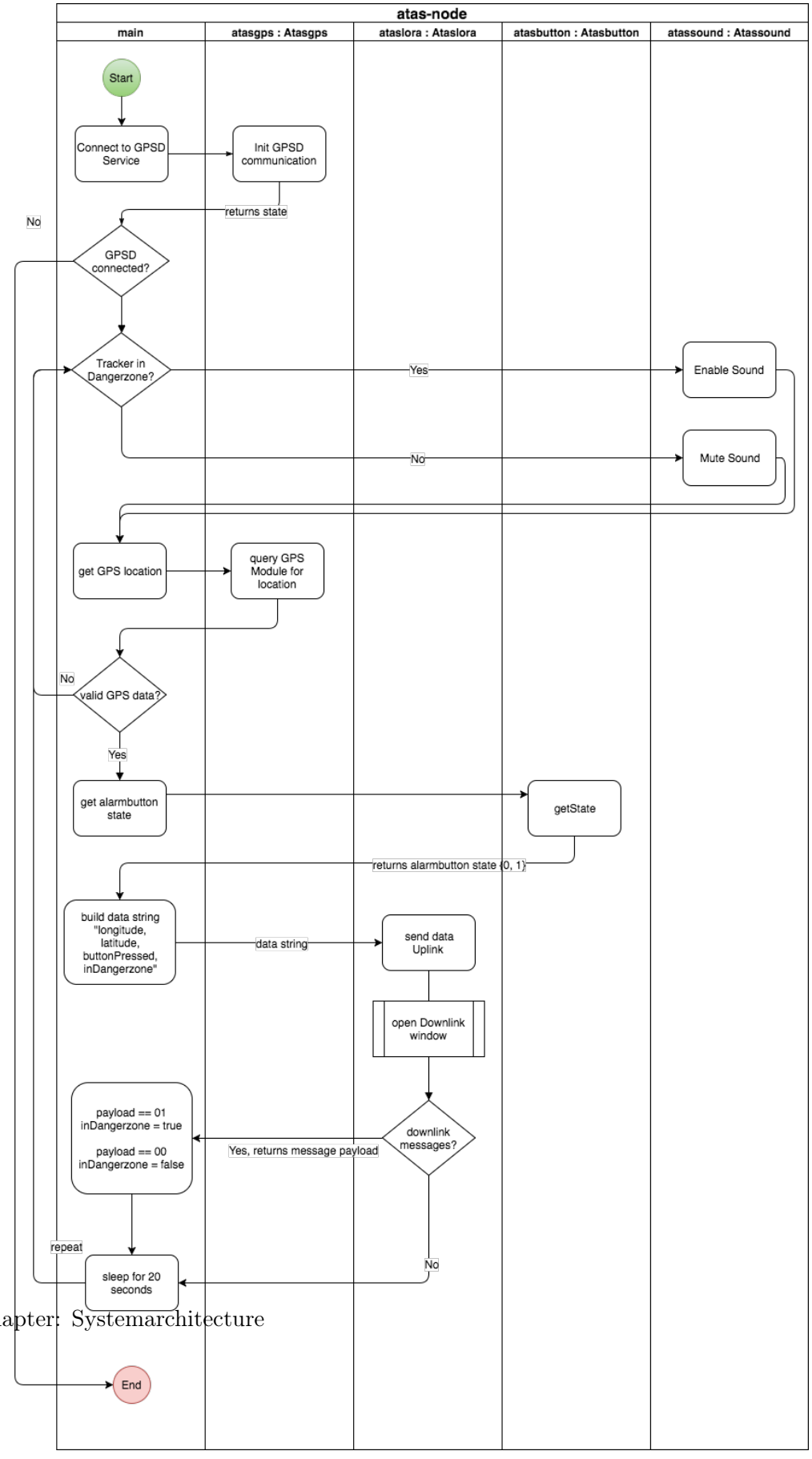
Functionality

For every task I created a separate class in c++.

- **Ataslora**
  Handles the communication with the LoRaWan Module. Used for sending and receiving data
- **Atasgps**
  Handles the communication with the GPSD Deamon. Queries the GPS location.
- **Atasbutton**
  Initialized the GPIO PIN to act as an Input. Reads the GPIO levels, High or Low.
- **Atassound**
  Controls the Piezo speaker. Enables or disables the alarmsound.

Process - Hole program

On the next page you are going to see a flow chart of the atas-node programm. The flowchart doesn't represent the actual code. Dummy syntax /code was used to make the chart as understandable as possible.


atas-node
main
atasgps : Atasgps
ataslora : Ataslora
atasbutton : Atasbutton
atassound : Atassound
Start
Connect to GPSD Service
Init GPSD communication
returns state
No
GPSD connected?
Tracker in Dangerzone?
Yes
Enable Sound
No
Mute Sound
get GPS location
query GPS Module for location
No
valid GPS data?
Yes
get alarmbutton state
getState
returns alarmbutton state {0, 1}
build data string "longitude, latitude, buttonPressed, inDangerzone"
data string
send data Uplink
open Downlink window
payload == 01 inDangerzone = true
payload == 00 inDangerzone = false
Yes, returns message payload
downlink messages?
repeat
No
sleep for 20 seconds
End

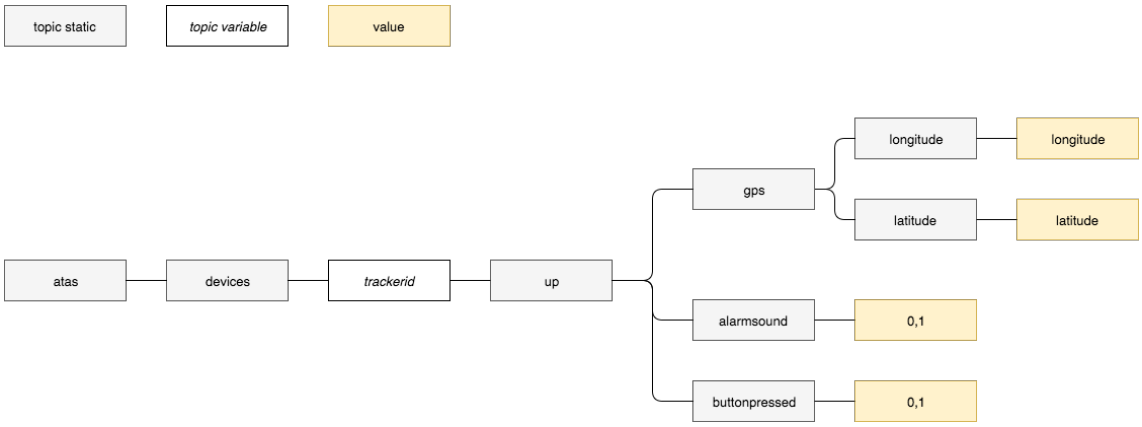
Chapter 3

Dataflow

3.1 MQTT

The following topic structure is used for the MQTT communication

3.1.1 Topics



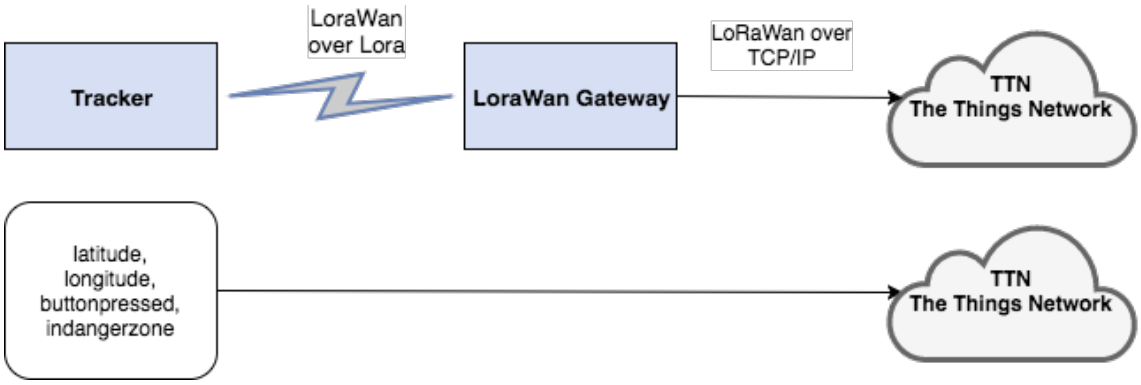
3.2 Tracker

This section will describe which data is send and received by the tracker

3.2.1 Tracker → TTN

The tracker sends messages to the TTN via LoRaWan gateway.
A message contains 4 values.

- Latitude: GPS latitude of the tracker
- Longitude: GPS longitude of the tracker
- Button pressed: If the switch has been set to on the value is 1 otherwise the value is 0.
- In Dangerzone: When a tracker is inside a dangerzone it will be informed by the atas-service (downlink). The tracker stores this informaton but also sends back the information to the TTN (uplink). This mechanism was developed to have an acknowledement from the trackerto check if he received the alarm signal. The Value is 1 if the tracker is in a dangerzone and 0 if the tracker is not inside a dangerzone.



Chapter 4

Conclusion

I had my concerns at the beginnen of this project as it required some know-how in electronic components, currenct flow and so on. As I'm an IT system engineer and not an electronics technician I always tried to avoid this topic, but im glad I confronted myself with it during this project.

During this project I designed a systemarchitecture, created a functioning prototyp and extended my kowledge about technologies like MQTT, LoRaWan and the The Things Network. I learned many interesting new things. Most importantly I created something which actually can be used in this world and be a help to somebody.

Looking back I should have done one thing differently. I should have planned my time better by creating a projects-timetable and stick to it. In the last couple weeks I worked very hard on this project while during the first half of the semester I only did the minimum.

This project is far from being completed. I see a huge potential in this kind of technology and would like to invest more time to bring this project to the next level. There are many components which should be improved and many aspects which should be researched and tested better. For example what is the maximum distance a signal can be transmitted? Does the weather have an impact on the signal? Does the signal travel through snow? And so on. There are many unanswered questions for which I saddly didn't have time to research.

Chapter 5

Attachments

5.1 Code

All code written for this projects hase been uploaded to github platform. No code will be attached to this documentation. Follow the links provided to access the code.

5.1.1 Atas-Webapp

`https://github.com/schmm2/atas-webapp`

5.1.2 Atas-Node

`https://github.com/schmm2/atas-node`

5.1.3 Atas-Service

`https://github.com/schmm2/atas-service`