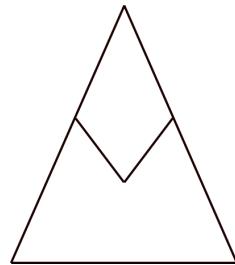




BERN UNIVERSITY OF APPLIED SCIENCES

BACHELOR THESIS
STUDIENGANG INFORMATIK

Alpinist Tracking & Alerting System



Autor:
Martin SCHMIDLI

Betreuer:
Mohamed MOKDAD

Experte:
Daniel VOISARD, BAKOM

Bern, 8. Januar 2018

Inhaltsverzeichnis

1 Einleitung	6
1.1 Projektidee	8
1.2 Vorarbeiten	8
1.3 Aufgabenstellung	8
1.4 Rahmenbedingungen	8
1.5 Technologien	9
1.6 Anwendungsfälle	9
2 Systembeschreibung - Grob	10
2.1 Grobe Systemarchitektur	10
2.2 Aufbau	11
3 Systembeschreibung - Detail	14
3.1 Tracker	16
3.2 Gateway	17
3.3 Broker	18
3.4 Logik	21
3.5 Webapplikation	22
4 Technologien	23
4.1 LoraWan & LoRa	23
4.2 The Things Network	31
4.3 MQTT	32
5 Prototyp	33
5.1 Begründung neuer Prototyp	33
5.2 Ablauf Prototyping	34
5.3 Ablauf Vorgehen	35
5.4 Evaluation Hardware	36
5.5 Zusammenbau Hardware	39
5.6 Kommunikation	41
5.7 Software	43
5.8 Finalisierung & Testing	48

6 Testing	49
6.1 Ablauf Vorgehen	49
6.2 Definition	50
6.3 Standort	51
6.4 Aufbau & Vorbereitungen	55
6.5 Messungen	58
7 Schlusswort	62
7.1 Verbesserungen bestehendes System	62
7.2 Ausblick ATAS	62
7.3 Rückblick Projekt	62
7.4 Software	64
7.5 Libraries	65

Management Summary

Viele Bereiche des öffentlichen Lebens wurden durch die digitale Revolution transformiert. Waschmaschinen werden intelligent, Roboter putzen das Haus, Autos fahren selbstständig. Das Thema IoT und Mobile Computing sind allgegenwärtig. Was aber passiert in der Bergwelt? Im Jahr 2016 kam es zu über 2800 Unfällen in den Schweizer Alpen. Viele Personen wurden verletzt, 178 endeten sogar tödlich [10]. Welche Fortschritte sind zu erwarten um in der Bergwelt das Risiko zu minimieren? Wie schreitet in der Bergwelt die Digitalisierung voran? Nur selten wird darüber gesprochen. Der Fokus der großen Konzerne, die im Bereich IoT und Mobile Computing aktiv sind, liegt auf den Bereichen in denen ein grosser Umsatz zu erwarten ist. Wäre es nicht auch wichtig in die Sicherheit in den Bergen zu investieren? Wie viele Unfälle könnten verhindert und wie viele Leben gerettet werden?

Genau bei diesem Punkt setzt das ATAS Projekt an. ATAS ist ein von mir erfundenes System welches auf der Basis von IoT Technologien versucht die Wahrscheinlichkeit für einen Unfall in den Bergen zu verkleinern. Das während dem Bachelor Thesis Vorprojekt (Projekt 2) erstellte Konzept, bildet die Grundlage für diese Thesis.

Der Bericht gliedert sich in 4 Bereiche. Im ersten Teil der Thesis wird die Funktionalität und der Aufbau des Systems genauer erläutert. Der Bericht startet mit einer groben Systembeschreibung und wird im Verlauf der Thesis detaillierter. Im Vordergrund stehen hier die Aufgabe der einzelnen Komponenten (Hard und Software) und deren Kommunikation untereinander. Im zweiten Teil geht es um die Technologien welche das System verwendet. Warum wurde diese gewählt und wie werden diese eingesetzt. Im dritten Teil wird der Aufbau eines neuen Hardware Prototypen beschrieben. Der im Vorprojekt aufgebaute Prototyp weist einige Schwachstellen auf und musste verbessert werden. Im vierten Schritt wird das ATAS System getestet. Im Fokus liegen hier die Neuerungen beim Prototypen sowie die Datenübertragung zwischen den Komponenten. Am Ende werde ich ein Fazit aus den Tests ziehen und bewerten, ob dieses System einen praktischen Nutzen bietet und Alltagstauglich ist.

Motivation

21.01.2017. Region Zweisimmen. Es war ein schöner Tag für eine Schneeschuhwanderung. Meine Kollegin und ich machten uns bereit. Wir schlüpften in die Schneeschuhe, packten die Skistöcke und zogen los. Meter für Meter kämpften wir uns den Weg zum Gipfel vor. Nach 3 Stunden hartem und schweißtreibenden Aufstieg gelangten wir schlussendlich zum Bergkamm. Die Aussicht war unser Lohn.



Abbildung 1: Motivation Alpen Panorama

Ich bin kein extrem Bergsteiger. Ich genieße die Schweizer Alpen, bin aber nie in hohen(3000m+) und gefährlichen Regionen unterwegs. Die Schneebedingungen während meiner Touren sind meist ideal. Die Gefahr ist minimal. Nur selten werden Themen wie Lawinen und daraus resultierendes Unfallverhalten in der Gruppe diskutiert.

An eben diesem Tag stellten wir uns die Frage: Was wäre wenn? Was wäre passiert wenn wir von einer Lawine verschüttet worden wären? Wer hätte uns gesucht, hätte uns überhaupt jemand gesucht. Falls meine Kollegin verschüttet wäre, wie lange würde es dauern bis die Rettungskräfte eintreffen würden? Wäre das Handynetz verfügbar gewesen? Und so weiter...

Diese Thematik beschäftigte mich noch lange. Als es schließlich darum ging ein Thema für das Bachelor Thesis Vorprojekt (Projekt 2) auszusuchen, wollte ich mich weiter mit diesen Fragen auseinandersetzen. Dank meinem Studium und der gewählten Fachrichtung 'Mobile Computing' hatte ich bereits ein, zwei Idee wie man diese Problematik angehen könnte. Das Projekt ATAS war geboren. Meine Vorstellung war es ein System zu erstellen, welches Personen in solchen Situation unterstützen, mehr noch solche Situationen verhindern sollte.

Danksagung

TODO PRIO 3

Kapitel 1

Einleitung

Ein Gipfel gehört dir erst, wenn du wieder unten bist - denn vorher gehörst du ihm.

Hans Lander, Italienischer Bergsteiger[9]

Die Berge sind eine faszinierende Landschaft. Viele Menschen gehen wandern, Ski fahren oder gehen Eisklettern. Die Anzahl der möglichen Aktivitäten ist schier grenzenlos. Immer wieder kommt es in dieser Idylle aber zu schweren Unfällen. Ausgelöst durch Steinschläge, Lawinen, Abstürze der Bergsteigenden/Alpinisten und vielen weiteren möglichen Ursachen. Der Schweizer Alpenclub (SAC) führt dazu eine Statistik [10]. Nachfolgend ein kleiner Auszug aus der Bergunfallstatistik 2016. Die Grafik zeigt eine Übersicht der Anzahl Situationen wo eine Rettungsdienst ausrücken musste. Zu den Rettungsdiensten gehört beispielsweise die REGA.

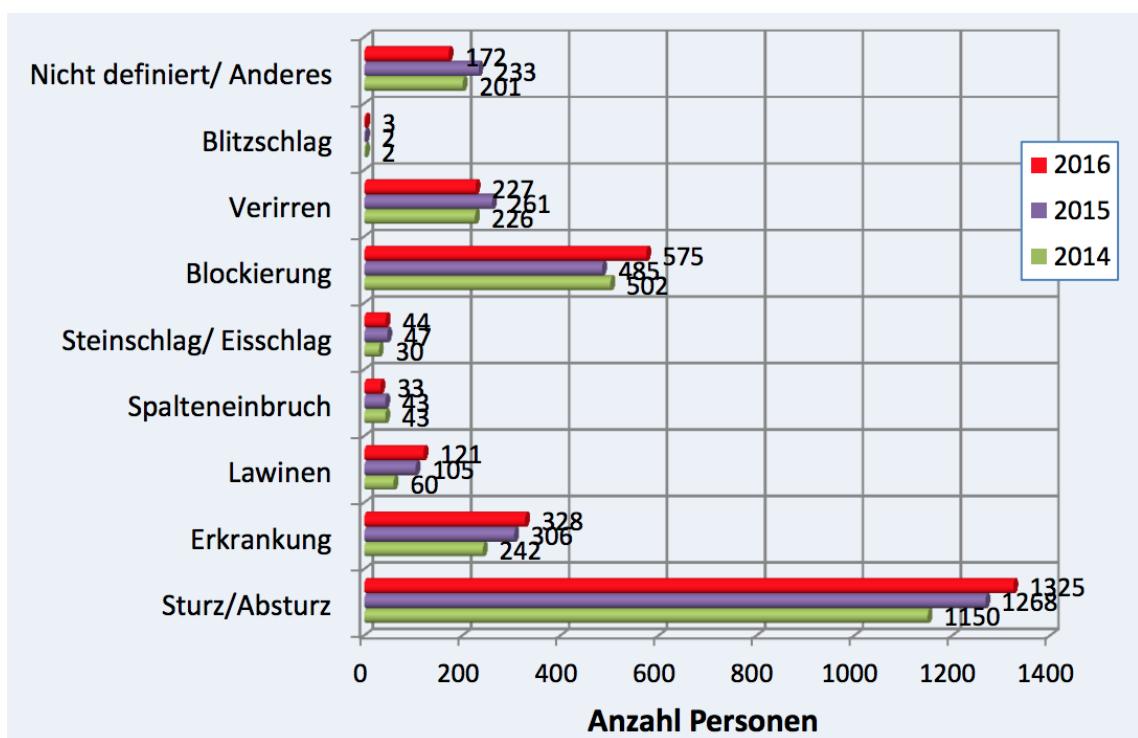


Abbildung 1.1: Bergunfallstatistik 2016 - Notfallsituationen nach Ursachen

Die Vorfälle sind nach Ursache gegliedert. Zusammengefasst ergeben sich 2828 Notsituationen, wovon 178 tödlich endeten.

1.1 Projektidee

Stellen Sie sich ein kleines mobiles Gerät vor, nachfolgend Tracker genannt, welches Skifahrern, Wanderer usw. abgegeben werden kann. Das Gerät sendet die Position der Person an einen Empfänger, nachfolgend Gateway genannt. Der Gateway wird bei der Talstation oder im nächsten Bergdorf montiert. Der Gateway sendet die empfangenen Daten der Tracker an eine zentrale Stelle irgendwo im Internet. Die Administratoren des Systems, beispielsweise die Rega, können schlussendlich über eine Webseite die aktuelle Position der Personen in den Bergen mitverfolgen und überwachen.

1.2 Vorarbeiten

Das Projekt 2 ist eine Projektarbeit welche im Laufe des Informatikstudiums absolviert werden muss. Diese Arbeit habe ich im vorhergehenden Semester (Frühlingssemester 2017) durchgeführt. Ziel dieses Projektes, war es das in der Einleitung beschriebene System technisch umzusetzen.

Diese Bachelor Thesis baut auf den Erkenntnissen und Resultaten aus dem Projekt 2 auf.

1.3 Aufgabenstellung

Ziel der Bachelorarbeit soll es sein, dass während dem Projekt 2 aufgebaute System intensiv zu testen und zu verbessern. Die Arbeit umfasst die folgenden Aufgaben.

- Der Tracker soll in realer Umgebung getestet werden bspw. während einer Wanderrung. Die Messdaten werden erfasst und analysiert.
- Der Tracker Prototyp soll verbessert werden. Die dazu erforderlichen Massnahmen werden während der Bachelorarbeit spezifiziert und umgesetzt. Am Ende der Arbeit soll eine klare Aussage gemacht werden können ob das erdachte System praxistauglich ist
- Die während dieser Arbeit geleisteten Arbeiten sollen in einem Bericht dokumentiert werden.
- Das Wissen rund um die verwendeten Technologien soll vertieft werden.

1.4 Rahmenbedingungen

Die Rahmenbedingungen dieses Projektes wurden gemeinsam mit dem Betreuer definiert

- Die Bachelorarbeit baut auf der Arbeit des Projekt 2 auf. Die erstellte Hard und Software wird weiterverwendet.
- The Things Network dient als Plattform für die LoRaWan Backen

- Als Programmiersprache soll C/C++ verwendet werden
- Auf den Einsatz eines Betriebssystems soll zugunsten des Energieverbrauchs auf dem Tracker Node verzichtet werden

1.5 Technologien

Sie als Leser denken nun vielleicht: "Das ist überflüssig. Ich habe doch mein Handy für so etwas? Schreib doch eine App!". Diese Aussage mag auf die touristischen Wander- und Skigebiete zutreffen. Der Empfang ist meistens wunderbar. Verlassen wir diese 'sicheren' Orte und bewegen uns aber in höheren Lagen, wird der Empfang mit dem Handy immer schlechter oder existiert gar nicht.

Aus diesem Grund mussten für dieses Projekt andere Technologien gefunden werden. In diesem Projekt werden folgenden Technologien eingesetzt: MQTT, LoRa, LoRaWAN, The Things Network

Die verwendeten Technologien und der Grund für deren Einsatz werden im Kapitel 'Technologien' genauer erklärt.

1.6 Anwendungsfälle

Dieses Abschnitt soll Ihnen erläutern, wie das ATAS System genutzt werden kann.

- Wenn in den Bergen eine Lawine ausgelöst wird, kann deren Position mit den Positionen der Tracker verglichen werden. Befindet sich ein Tracker in der Gefahrenzone, kann die Rettungsmannschaft sofort reagieren und ausrücken. Dieser Prozess läuft sofort ab und kann dabei die Überlebenschance der Opfer erhöhen.
- Wurden Personen von der Lawine begraben und der Tracker hat überlebt, könnte das Gerät weiterhin die Position senden. Kombiniert mit modernen Lawinensuchgeräten können die Einsatzkräfte gezielter nach Überlebenden suchen. Wenn keine Übertragung mehr möglich ist, wissen die Überwacher zumindest den letzten Aufenthaltsort. Das ATAS System hat nicht das Ziel Lawinensuchgeräte zu ersetzen, es ist eher als Ergänzung zu verstehen.
- Bewegt sich Tracker auf eine Gefahrenzone zu, beispielsweise ein Gebiet mit erhöhter Steinschlaggefahr, könnte die Person frühzeitig davor gewarnt werden.
- Ist es zu einem Unfall gekommen, kann der Alpinist mittels Tracker ein Notsignal absetzen. Dazu muss der Benutzer nur auf den Notfallknopf drücken.

Kapitel 2

Systembeschreibung - Grob

Um die nachfolgenden Kapitel zu verstehen, ist es sehr wichtig, sich mit der im Projekt 2 erstellten Systemarchitektur und die Terminologie vertraut zu machen. Die Architektur wurde im Vergleich zum Projekt 2 um einige Komponenten ergänzt.

2.1 Grobe Systemarchitektur

Dieser Abschnitt bietet Ihnen einen groben Überblick über die Benutzer, die Komponenten und deren Beziehung innerhalb des ATAS Systems. Alle Einheiten werden auf den nachfolgenden Seite detailliert beschrieben.

2.2 Aufbau

2.2.1 Diagramm

Grobes Schema des ATAS Systems. Auf der nachfolgenden Seiten sind die einzelnen Komponenten im Detail beschrieben.

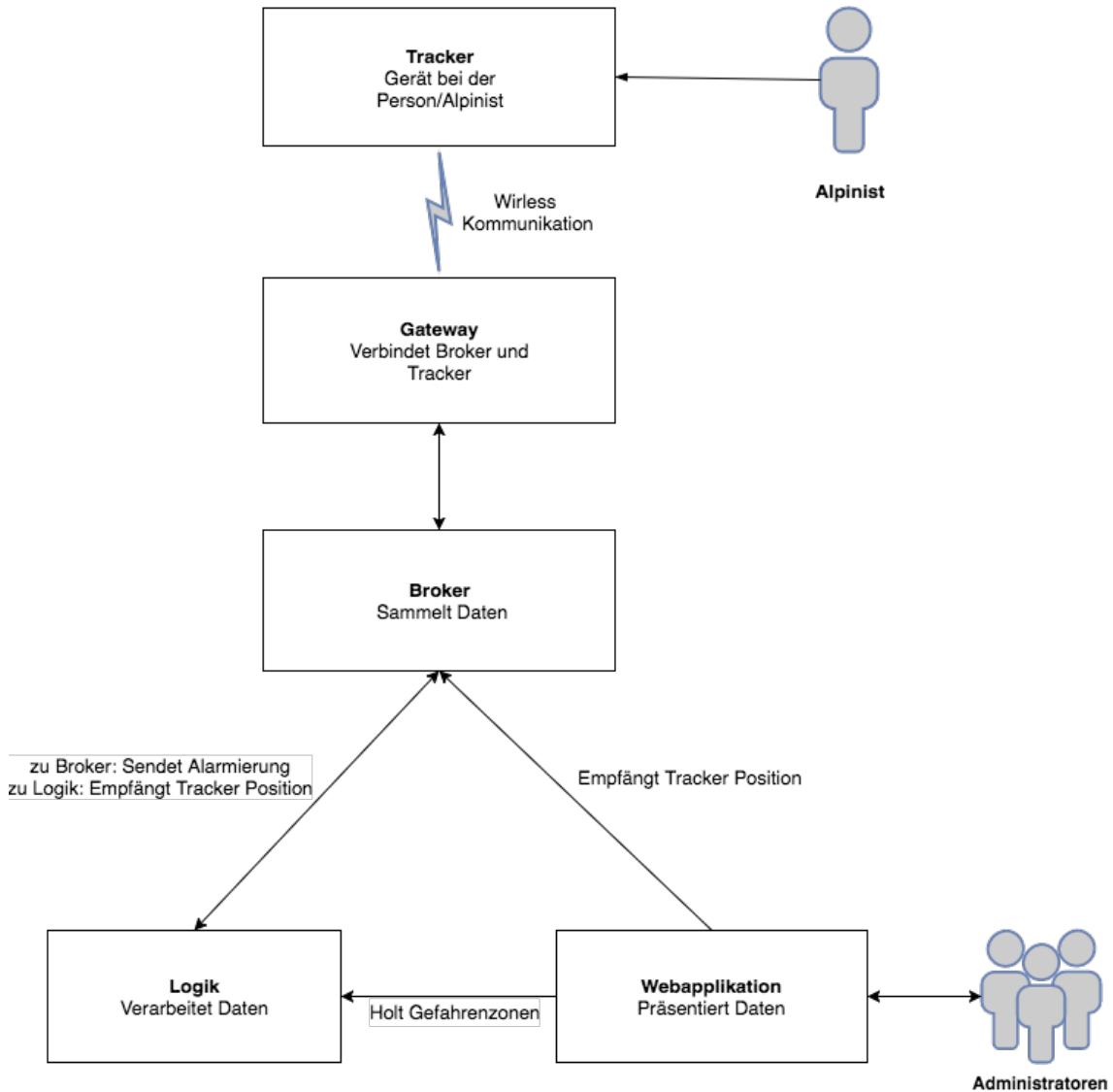


Abbildung 2.1: Atas Schema Grob

2.2.2 Benutzer

Die nachfolgenden Gruppen wurden als Benutzer des Systems identifiziert.

Aplinist

Alpinist wird als Generalisierung für Personen, welche sich in den Bergen aufhalten verwendet. Dazu gehören bspw. Wanderer, Skifahrer, Bergbauern usw.

Überwacher

Rettungsdienste wie bspw. die Rega oder die AirGlacier. Spitäler oder die lokalen Tourismusbehörden.

2.2.3 Komponenten

Das ATAS System besteht aus den nachfolgenden Komponenten. Komponenten sind als Hardware und Software zu verstehen.

Tracker

Stellen Sie sich ein kleines mobiles Gerät vor, nachfolgend genannt Tracker. Der Alpinist trägt den Tracker bei sich bspw. in einem Rucksack.

Der Tracker verfügt über ein Display, Taste, GPS Modul, Kommunikationsmodul und ein Lautsprecher.

Gateway

Ein Gerät welches mit den Trackern bidirektional kommuniziert. Der Gateway wird an einem Ort nahe den Bergen installiert. Als mögliche Orte kommt ein Dorf im Tal oder eine Talstation in Frage.

Der Gateway senden die empfangenen Daten an eine zentrale Datenmanager, nachfolgend **Broker** genannt.

Broker

Sammelt und speichert Daten. Der Broker kommuniziert mit den Gateways. Zusätzlich bietet der Broker ein Interface zum Abfragen und senden von Daten.

Webapplikation

Die Notfalldienste resp. Überwacher nutzen eine Webapp zum interagieren mit dem ATAS System. Die Webseite bietet folgende Funktionalitäten

- Zeigt die Position der Tracker auf einer Karte an

- Mittels der Webapp können Gefahrenzonen hinterlegt werden.
- Zeigt an ob sich ein Tracker in einer Gefahrenzone aufhält.

Business Logik

Die BusinessLogik umfasst Software welche berechnet, ob sich ein Tracker in einer Gefahrenzone aufhält. Wenn sich der Tracker in einer Gefahrenzone aufhält, senden das System eine Alarm an den Tracker. Das Signal gelangt via Broker und dem Gateway zum Tracker.

2.2.4 Begriffe

Gefahrenzonen

Wie in der Einleitung beschrieben, stellt der Schweizer Alpenclub Informationen zu Rettungseinsätzen zur Verfügung. Aufgrund der Statistiken wurden die folgenden Gefahren identifiziert vor welchen das ATAS System informieren soll.

- Lawinen
- Steinschlag
- Spalte, bspw. Gletscher
- Schneestürme

Kapitel 3

Systembeschreibung - Detail

Die vorhergegangene grobe Architekturbeschreibung gibt einen guten ersten Überblick über das System. Technische Details zu den verwendeten Technologien oder den genauen Fluss der Kommunikation ist nicht ersichtlich. In diesem Abschnitt soll nun intensiv auf diese Thematik eingegangen werden.

Auf der nachfolgenden Seite finden Sie ein Schema welches einen genauen Überblick über das System liefert.

Die Komponenten werden in 2 Kategorien eingeteilt. Komponenten mit einem blauen Kästchen wurden im Rahmen dieser Arbeit selbständig programmiert. Gelbe Kästchen markieren einen bestehenden Service welcher für den Aufbau des Systems implementiert wurde.

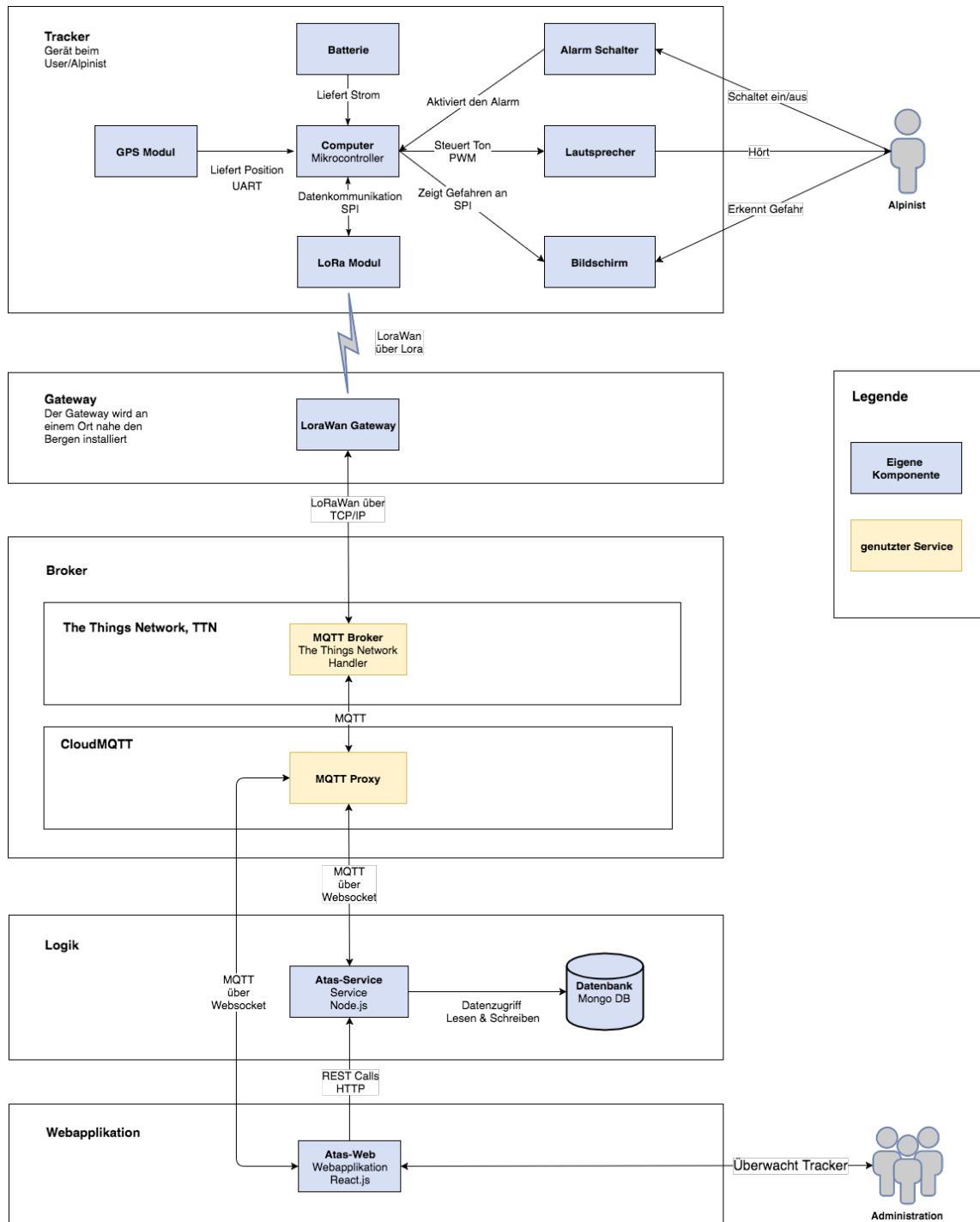


Abbildung 3.1: Systemübersicht Detail

3.1 Tracker

Wie in der untenstehenden Grafik ersichtlich, besteht der Tracker aus mehreren Komponenten/Modulen. Welche Komponenten verbaut und wie diese untereinander kommunizieren, wird im Kapitel 'Prototyp' genauer behandelt.

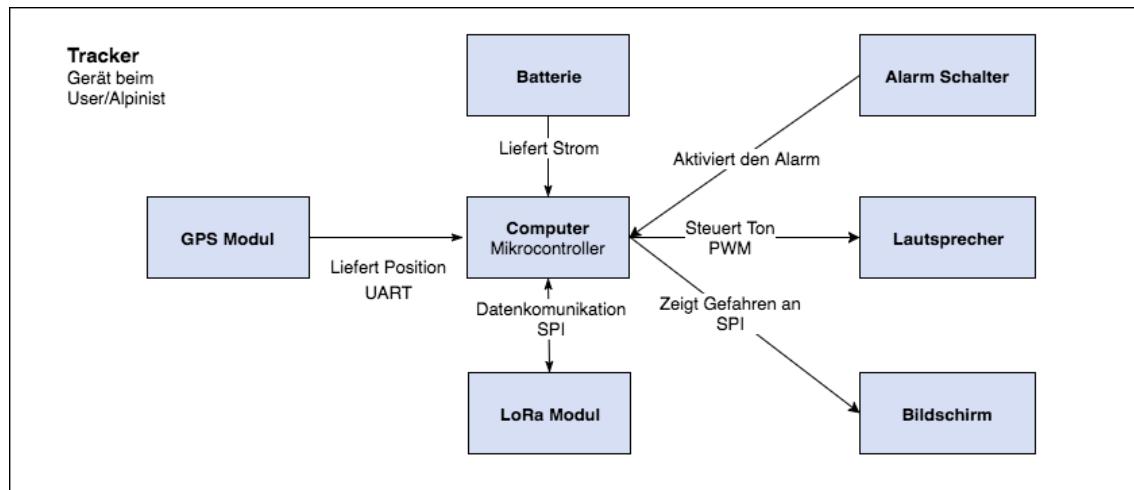


Abbildung 3.2: Tracker Detail

- **Schalter:** Mit dem Druck auf den Knopf können die Überwacher über eine Notsituation aufmerksam gemacht werden. Bspw. Wenn der Alpinist einen Unfall hatte und nun bewegungsunfähig ist.
- **Lautsprecher:** Über den Lautsprecher kann der Alpinist über eine Gefahrenquelle mit einem akustischen Signal aufmerksam gemacht werden. Bspw. Wenn sich der Alpinist in einem Bereich am Berg mit erhöhter Gefahr für Lawinen aufhält. Der Lautsprecher dient zur Information, **dass ein Problem besteht**.
- **Display:** Über ein Display können mehr Informationen zum Tracker und den Gefahrenzonen angezeigt werden. Das Display dient zur Information, **was für ein Problem besteht**.
- **GPS Modul:** Der Tracker verfügt über ein GPS Modul. Mit dem GPS Modul kann die Position des Tracker auf der Erde ermittelt werden.
- **Lora Modul:** Mittels Lora Modul können Daten an einen Empfänger, nachfolgend **Atas-Gateway** genannt, gesendet werden.

3.2 Gateway

Der Gateway verbindet die Tracker mit dem TTN Netzwerk. Er sendet Daten zu den Trackern (Downlink) und empfängt Daten von den Trackern (Uplink). Was genau das TTN für eine Rolle übernimmt, wird später im Detail beschrieben.

3.2.1 Diagramm

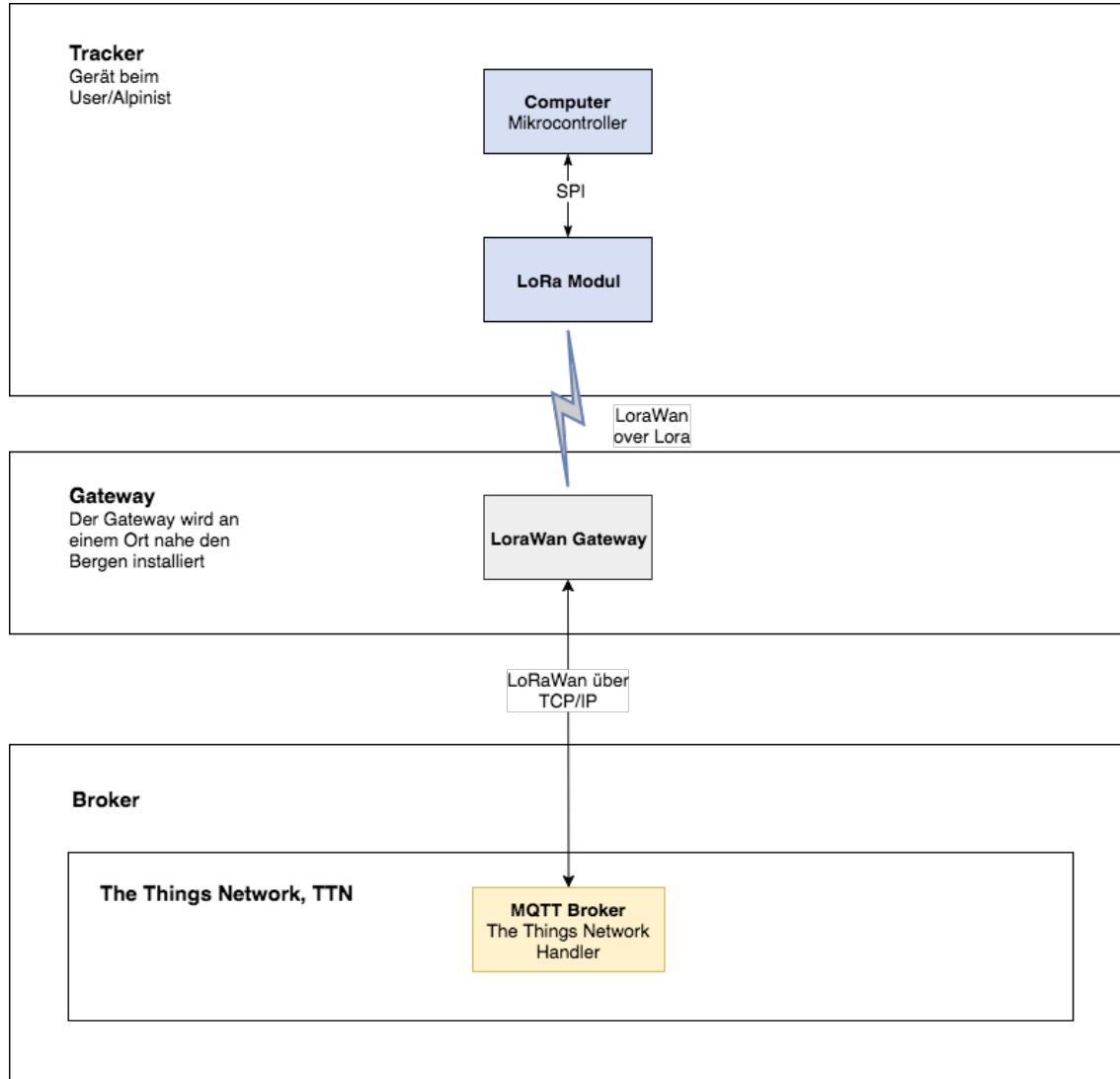


Abbildung 3.3: Gateway Detail

3.3 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.

3.3.1 The Things Network (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

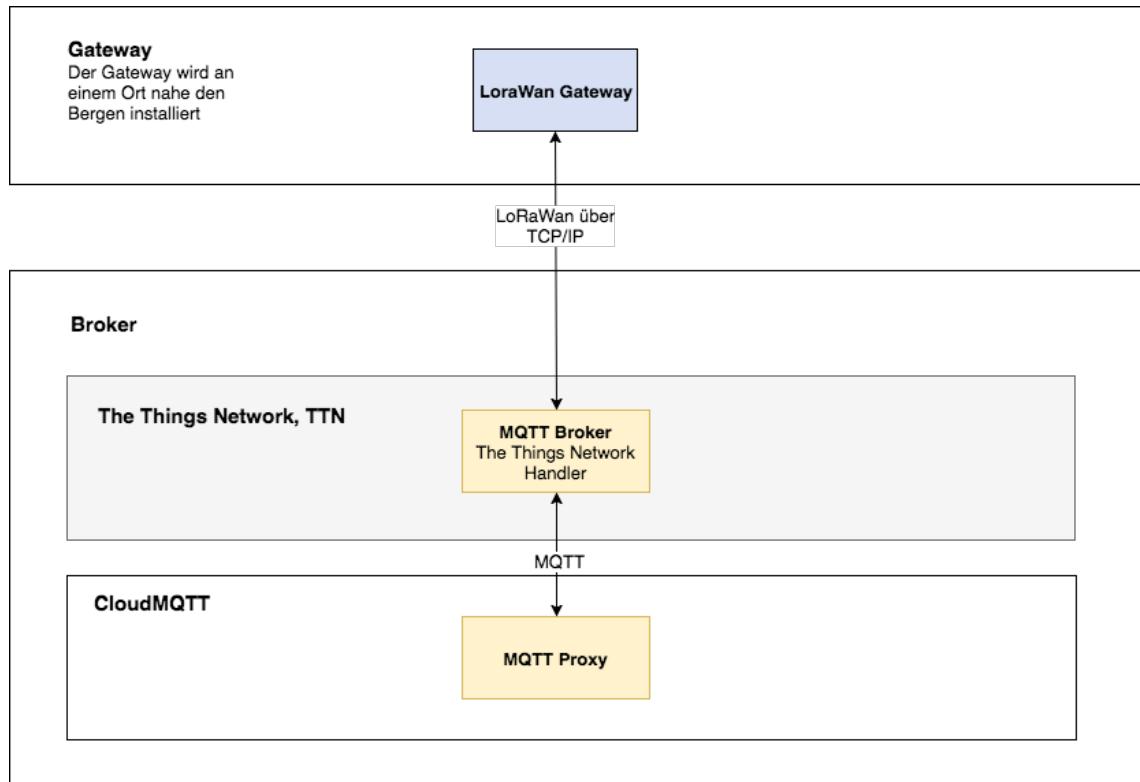


Abbildung 3.4: TTN Detail

3.3.2 MQTT

Für das ATAS Projekt wurde die nachfolgende MQTT Struktur verwendet. Einige Topics sind durch das TTN fest vergeben und konnten nicht selber gewählt werden.

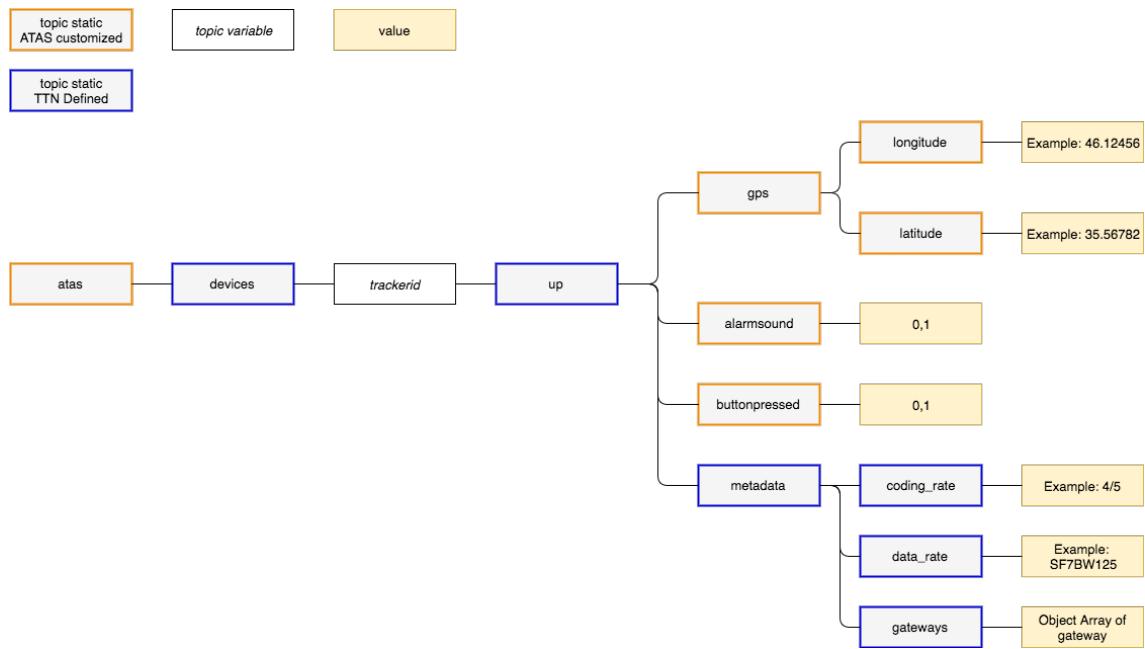


Abbildung 3.5: MQTT Struktur

3.3.3 CloudMQTT

Der MQTT Broker des TTN Netzwerkes bietet nur die Möglichkeiten die Daten via MQTT abzurufen. Ein Zugriff via MQTT über Websockets wird nicht unterstützt. Im Internet fand ich recht schnell den Service 'Cloud' MQTT. CloudMQTT spiegelt den TTN MQTT Broker, bietet aber zusätzlich ein WebSocket Interface. Atas-Web und Atas-Service können nun via CloudMQTT Daten an die Tracker senden und empfangen.

Alle Daten welche zu CloudMQTT publiziert werden, werden nun auch automatisch zum TTN MQTT Broker publiziert (Published). Alle MQTT Topics welche vom CloudMQTT abonniert werden, werden auch vom TTN MQTT Broker abonniert (Subscribe).

Diagramm

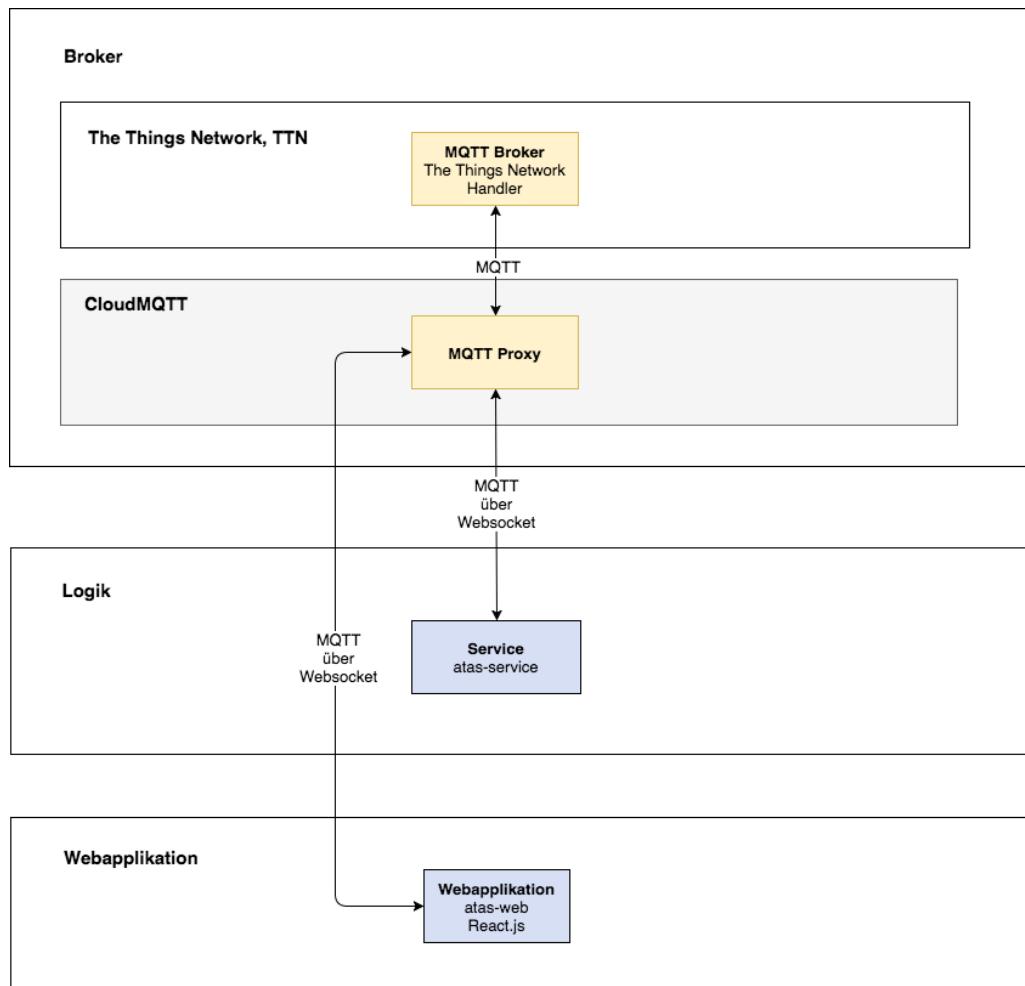


Abbildung 3.6: TTN Detail

3.4 Logik

3.4.1 Atas-Service

TODO, PRIO 2

3.5 Webapplikation

3.5.1 Atas-Web

TODO, PRIO 2

Kapitel 4

Technologien

In diesem Kapitel möchte ich auf die eingesetzten genauer Technologien eingehen. Ich werde beschreiben wofür diese eingesetzt werden und weshalb ich mich gerade für diese Technologien entschieden habe.

4.1 LoraWan & LoRa

Wer sich mit LoRa beschäftigt stößt früher oder später auf den Begriff LoRaWan. LoRaWan ist ein Protokoll welches auf LoRA als Übertragungsart aufsetzt. Diese beiden Begriffe werden hier gemeinsam analysiert. Um das Projekt durchzuführen, musste ich mir mehr Wissen zum Thema LoRA & LoraWan aneignen. Das im Projekt 2 aufgebaute System ist lauffähig, verwendet aber gewisse Standardeinstellungen welche ich einfach übernommen habe. Das Wissen, ob und wie man das System optimieren könnte war nicht vorhanden.

4.1.1 Vorteile

- Durch bereits eingebaute Sicherheitsmechanismen in LoRaWan sind die übertragenen Daten geschützt
- Durch den kleinen Energiebedarf halten Batteriebetriebene Geräte länger
- Die Kosten für die Installation eines LoRaWan Gateways sind gering +- 300.-
- Grosse Reichweite in nicht bebauten Gebieten (+-15km)
- Geringe Störanfälligkeit durch LoRa Chrip

4.1.2 Nachteile

4.1.3 Verbreitung

TODO, PRIO 3 SWISSCOM

4.1.4 Airtime & Duty Cycle

TODO, PRIO 2

Spreading Factor

TODO, PRIO 2

4.1.5 Endgerät Verbindungsmethoden

OTAA (Over-The-Air-Activation) und ABP (Activation By Personalization) sind Verbindungsmethoden für LoraWan Endgeräte in einem LoraWan Netzwerk.[15]. Die verwendete Methode definiert also wie sich ein Endgerät mit dem LoraWan Netzwerk verbindet.

Die beiden Methoden bieten verschiedene Vor und Nachteile.

ABP

ABP ist das simplere aber **weniger sichere** Verfahren. Die Adresse DevAddr sowie die Schlüssel NetSKey und des AppSKey werden für jedes Endgerät **einmalig** generiert und fix auf dem Endgerät hinterlegt. Gelingt es einem Angreifer sich Zugang zum Endgerät zu verschaffen, kann er die Schlüssel stellen, ein zweites Gerät am Netzwerk registrieren, die Identität des Originals annehmen und damit die Daten verfälschen.

Das Gerät muss sich nicht am Netzwerk anmelden. Das Endgerät kann direkt Daten senden. Empfängt ein Gateway die Daten, werden die Keys geprüft und die Kommunikation entsprechend an oder abgelehnt.

OTAA

Das Gerät muss sich am Netzwerk anmelden. Dieser Vorgang wird auch Join Prozedur genannt. Die DevAddr sowie die Keys NetSKey und AppSKey werden bei jeder Aktivierung des Gerätes **neu generiert** und an das Endgerät übertragen. Dieses Verfahren ist sicherer. Da es sich hier um eine bidirektionale Kommunikation handelt, müssen die Komponenten (Gateway & Endgerät) Downlinks unterstützen.

Das Installieren (Deployment) von Endgeräten wird vereinfacht. Das generieren vom AppSKey und NetSKey pro Endgerät entfällt.

Die nachfolgende Grafik zeigt auf, wie der Kommunikationsaufbau via OTAA abläuft [15].

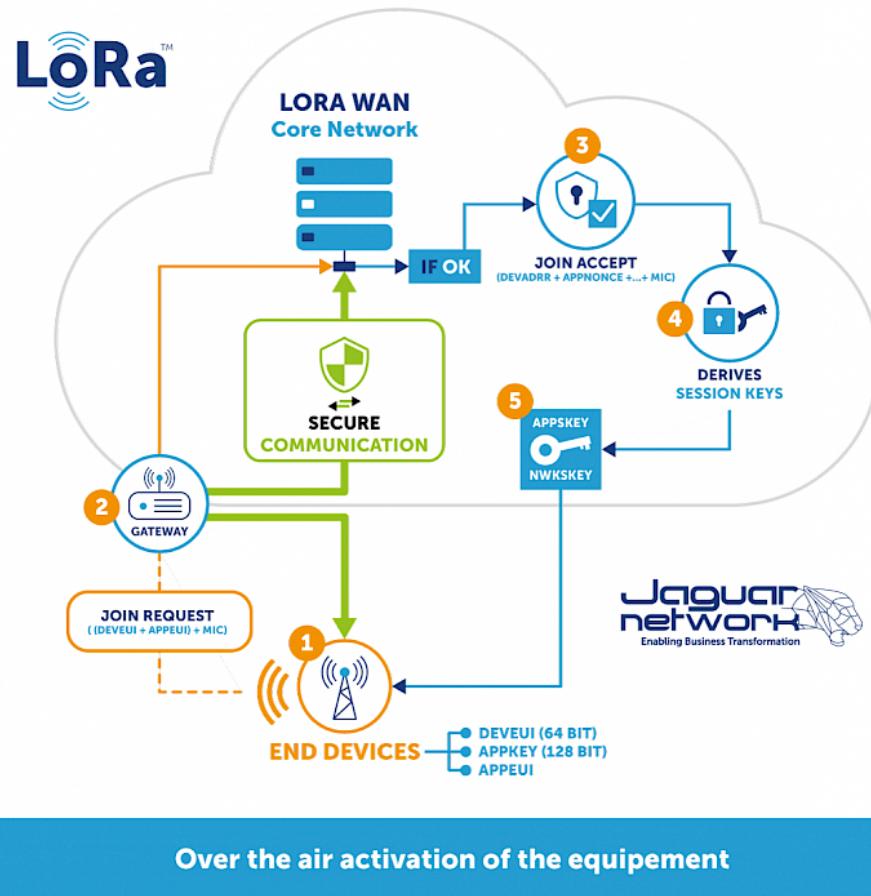


Abbildung 4.1: OTAA Ablauf

1. Gerät sendet einen Join Request.
2. Die Gateways empfangen die Anfrage
3. Das LoRaWan Netzwerk bspw. TTN prüft die Angaben
4. Session Keys werden generiert
5. Keys werden an das Endgerät gesendet und für die zukünftige Kommunikation verwendet

Schlussfolgerung

Obschon die Sicherheit während diesem Projekt nicht im Vordergrund steht, ist es sicher zukunftsorientierter direkt OTAA einzusetzen. Systeme welche zum Schutz von Personen eingesetzt werden, müssen so sicher wie nur möglich konzeptioniert sein.

Sowohl Endgerät als auch Gateway unterstützen Downlinks, damit gibt es keine technische Hürde der den Einsatz von OTAA verhindern würde. Aus diesem Grund habe ich während dem Projekt die Verbindungsmethodik von ABP auf OTAA umgestellt und konnte damit die Sicherheit des Systems verbessern.

4.1.6 Frame Counters

Ein Endgerät besitzt zwei Zähler (FCntUp, FCntDown). Die Zähler werden bei einer Downlinkmessage (FCntDown) resp. Uplinkmessage (FCntUp) erhöht (+1).

Wird eine Reply Attacke durchgeführt d.H. ein Paket nochmals gesendet, wird dieses Paket vom System verworfen. Dies geschieht deshalb, weil das System bereits eine Nachricht mit dem gleichen Framecounter erhalten hat.

Diese zusätzliche Information wird mir beim Testen der Übertragung überaus nützlich sein. Anhand der Nummerierung, kann sehr schnell erkannt werden, ob ein Paket nicht sauber empfangen werden konnte. Bspw. Erhalten wir die Pakete 4,5,7 → Paket 6 wurde nicht korrekt übertragen.

4.1.7 Geräteklassen

LoRaWan Endgeräte werden in drei Geräteklassen unterteilt [4]. Alle Geräte unterstützen eine bidirektionale Kommunikation. Es können also Daten zum Gateway gesendet wie auch empfangen werden.

Klasse A

Alle Geräte der Klasse B und C haben die Funktionalität von Klasse A implementiert. Das Endgerät kann immer Senden. Für den Empfang der Daten gibt es bestimmte Regeln. Nach einem Senden öffnet das Endgerät zwei sogenannte Download Receive Fenster. Während dieser Zeit können Daten Empfangen werden.

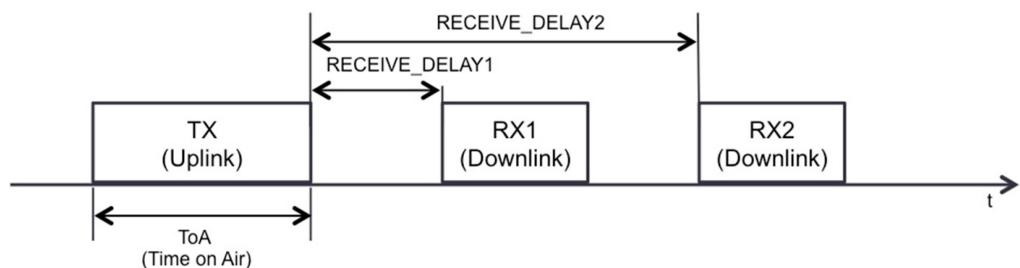


Abbildung 4.2: LoRaWan Class A

TODO eigenes BILD

Werden Daten außerhalb dieses Zeitfensters geschickt, werden diese nie vom Endgerät empfangen.

Klasse B

Ein Klasse B Gerät kann zusätzliche Download Receive Fenster zu definierten Zeiten öffnen. Der Gateway sendet ein Datenpaket (Beacon) an die Endgeräte um die Zeiten zu synchronisieren. Sind die Zeiten synchronisiert, weiß das LoRaWan Netzwerk ganz genau wann ein Endgerät bereit ist Daten zu empfangen.

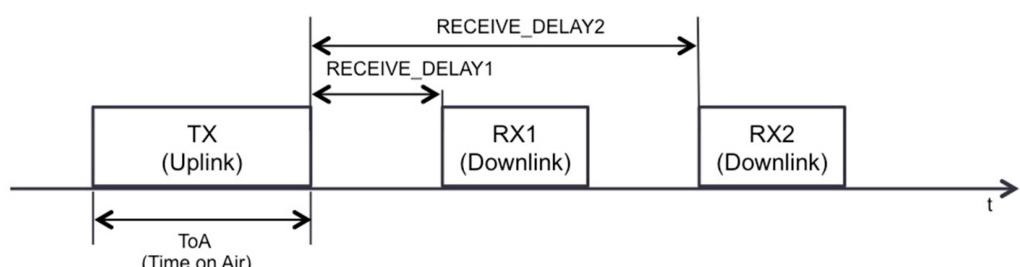


Abbildung 4.3: LoRaWan Class B

TODO eigenes BILD

Klasse C

Klasse C Geräte haben ein nahezu permanent geöffnetes Receive Zeitfenster. Nur während einem Sendevorgang wird dieses kurzzeitig geschlossen. Durch diese Eigenschaften ist der Energieverbrauch von Klasse C Geräten im Vergleich zu A & B sehr viel größer.

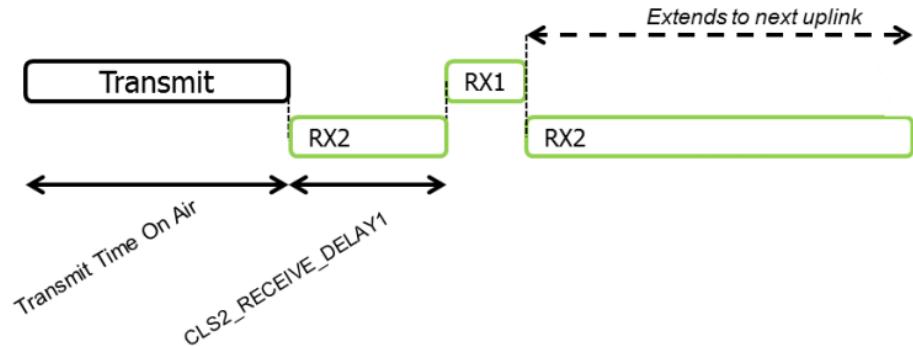


Abbildung 4.4: LoRaWan Class C

TODO eigenes BILD

Entscheid

Klasse B und A werden vom TTN System noch nicht unterstützt. Der Energieverbrauch von Klasse A Geräte ist im Vergleich von B & C geringer. Diese Tatsachen machen den Entscheid sehr einfach. Für die ATAS Tracker werden wir also die Klasse A verwenden.

4.1.8 Adaptive Data Rate (ADR)

Ist Adaptive Data Rate aktiv (ADR), entscheidet das LoraWan Endgerät selbstständig ob und wie die Kommunikation optimiert werden soll. Gemäss TTN [16] sollte ADR nur für statische Nodes oder für mobile Nodes zur Erkennung von Stops (keine Bewegung) verwendet werden. Gemäss TTN ist ADR also nicht unbedingt für die Atas-Tracker geeignet. Die Tracker sind ja meistens in Bewegung bspw. durch Laufen, Ski fahren usw. Dennoch wurde mein Interesse geweckt und ich wollte genauer wissen warum dieses Feature nicht empfohlen wird.

Sobald ein LoraWan Endgerät dem Netzwerk mitteilt, dass es ADR verwenden möchte, beginnt das TTN Daten aufzuzeichnen. Die letzten 20 Messdaten die der Node gesendet hat werden analysiert. Ist der Node nahe dem Gateway wird der Spreadingfactor (SF) heruntergeschraubt bspw. von SF10 auf SF7. Damit wird Airtime und Energie gespart.

Entfernt sich der Node vom Gateway wird SNR und damit RSSI schlechter. Das System erhöht nun automatisch den SF um eine bessere Übertragung sicherzustellen.

Der Einsatz von ADR macht im ATAS System meiner Meinung nach wenig Sinn. Dafür gibt es zwei Gründe

- Die ADR Kommunikation generiert einen Overhead [5]. Die ADR Kommunikation benötigt jeweils zwei Uplink und eine Downlink Message.
- In der Zeit welche zwischen zwei Uplinks vergeht, kann sich die Umgebung drastisch verändern. Bspw. Der Wanderer hat direkten Sichtkontakt mit dem Gateway. Das System reduziert den Spreadingfactor. 2 Minuten später befindet er sich in einem Waldstück mit einem schlechteren Empfang. Die nun gesendeten Uplink erreichen möglicherweise das Ziel nicht.

Meine Empfehlung wäre die Entwicklung eines eigenen ADR Systems. Der SF ist grundsätzlich hoch (SF11-SF12) zu halten um eine Übertragung der Daten zu garantieren. Mittels Auswertung der letzten GPS Punkte könnte das System die Geschwindigkeit der Person erkennen. Bewegt er sich langsam (laufen), könnte die Zeit zwischen den einzelnen Uplinks erhöht werden. Damit sparen wir Energie. Bewegt er sich schneller (Ski Fahren) kann die Zeit reduziert werden.

4.1.9 Regulation

Für die Schweiz und in vielen Teilen Europas gilt für die Nutzung von LoRa die Regelung ERC/REC 70-03 [6]. Dieses Dokument beschreibt die verwendbaren Frequenzen und die maximale Übertragungsleistung (dbm). Während den Tests wurden diese Regulation eingehalten. Die maximale Sendeleistung von 25mW resp. 14dbm wurde nicht überschritten.

4.2 The Things Network

TODO PRIO 2

4.3 MQTT

Durch die Wahl vom TTN Netzwerk wurde mir die Datenübertragungsart MQTT quasi aufgezwungen. Wird der TTN Handler nicht selber verwaltet, ist der Zugriff auf die Daten (API) nur per MQTT möglich [3]. Die Unterstützung von AMQP ist geplant aber noch nicht verfügbar. Das Protokoll war für mich nicht neu. Während dem Studium konnte ich bereits positive Erfahrung mit MQTT sammeln. Das Protokoll ist einfach zu verstehen und hat sich sehr einfach in mein System eingliedern lassen. Die fehlende WebSocket Unterstützung konnte mit wenig Aufwand über CloudMQTT realisiert werden.

Kapitel 5

Prototyp

Das Bachelorarbeit wird in 2 Hauptaufgaben aufgeteilt. **Testing** und das erstellen eines zweiten Prototyps nachfolgend genannt **Prototyp 2**. Dieses Kapitel beschreibt, wie ich beim des Prototypen 2 vorgegangen bin.

5.1 Begründung neuer Prototyp

Es gibt diverse Gründe für diesen Schritt

1. Auf dem Prototyp 1 wird ein vollwertiges Betriebssystem (Raspian, Linux Derivat) eingesetzt. Einige Funktionen bspw. das Lesen der GPS Daten werden vom System sehr vereinfacht. Dies ermöglichte mir einen raschen und unkomplizierten Aufbau der Software. Obschon ein solches 'High Level' OS eine grosse Hilfe in der Entwicklung darstellt, ist es in der Industrialisierung der Lösung eher hinderlich. Der Einsatz eines OS hat diverse Nachteile
 - Vom OS selbst benötigen wir für den Betrieb der Atas-Node Software nur einen Bruchteil der Funktionalität. Speicher, Memory und Systemperformance wird für Systemprozesse verschwendet.
 - Durch die hohe Auslastung des Systems werden stärkere Prozessoren benötigt. Der Energieverbrauch der Lösung steigt.
 - Mehr Software bedeutet Zwangsläufig mehr Fehlerquellen. Obschon das eingesetzte System (Raspian) als Stabil gilt, kann es aufgrund der Softwarekomplexität zu Fehlern kommen.
 - Durch den Einsatz eines OS reduzieren wir die Sicherheit des ganzen Systems. Je mehr Interfaces ein System anbietet, desto anfälliger wird es für den unerlaubten Zugriff.
2. Der Prototyp 1 bietet und viele Anschlussmöglichkeiten. Bspw. für ein Display, USB, oder eine RJ45 Buchse. Funktionen die wiederum welche Entwicklung stark vereinfachen, aber den Energieverbrauch erhöhen und die Grösse des Systems unnötig vergrößert.

- Die Verbaute Komponenten sind nicht für eine Industrialisierte Lösung geeignet. Man könnte sogar von einem 'Bastelprojekt' sprechen. Die Komponenten hindern das Projekt daran eine professionelle Form anzunehmen.

5.2 Ablauf Prototyping

Aus meiner Sicht bildet während der Arbeit zu konstruierende Prototyp 2 nur einen Zwischenstufe bis zu Finalen Version. Eine Visualisierung meiner Vorstellung:

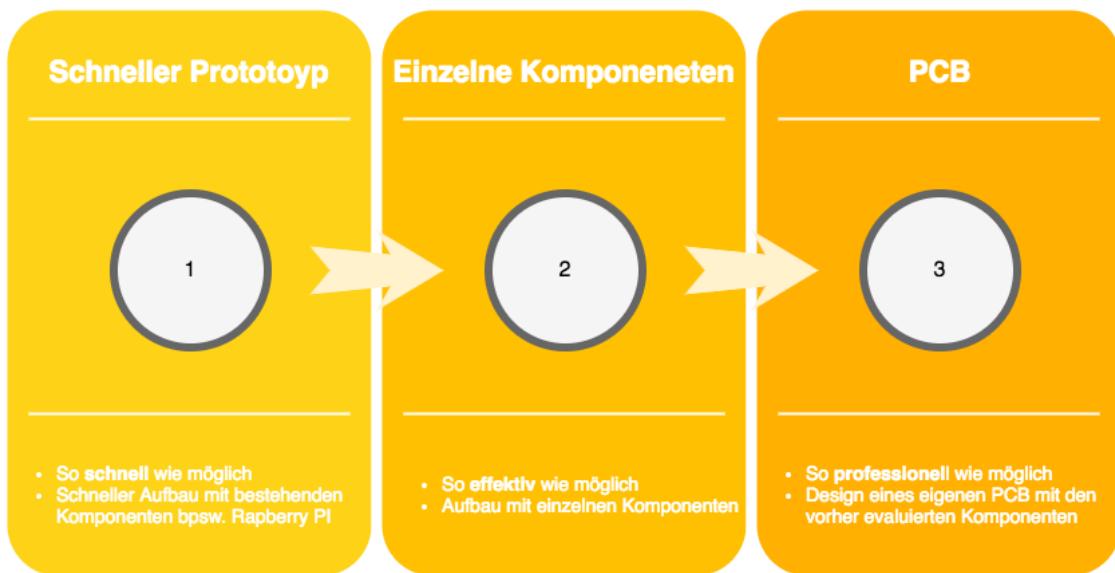


Abbildung 5.1: Prototyping Ablauf

Mein Ziel ist es mit dem Umstieg auf anderen Komponenten, den Grundstein für den Prototypen 3 zu legen. Der Prototyp 3 ist nicht Bestandteil dieser Arbeit. Prototyp 3 bildet die finale Prototyp Version. Die Einzelnen Komponenten vom Prototyp 2 werden auf einen eigens designeten PCB installiert. Durch diesen Schritt würde sich das System in Sachen Platzbedarf nochmals massiv verbessern.

5.3 Ablauf Vorgehen

Der im Projekt 2 erstellte Prototyp wurde mit sehr simplen vorgefertigten Elektronikkomponenten umgesetzt. Ziel dieser Arbeit war es in möglichst kurzer Zeit einen Prototypen aufzubauen. Während dieser Arbeit sollen nun neue Komponenten evaluiert und ein zweiter Prototyp erstellt werden. Der Ablauf dieser Phasen sieht wie folgt aus:

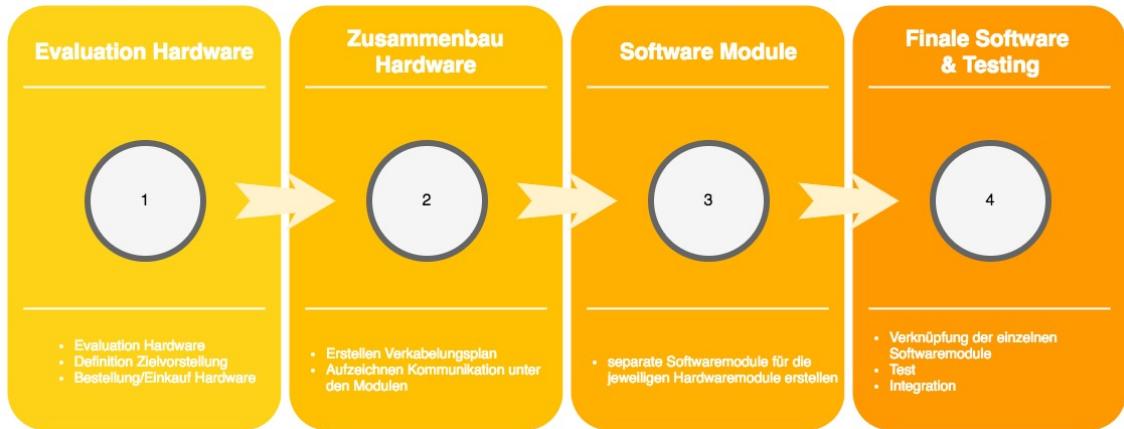


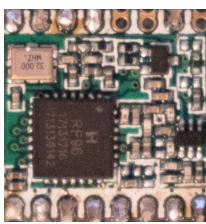
Abbildung 5.2: Ablauf für den Aufbau des zweiten Prototyps

Die einzelnen Schritte werden auf den kommenden Seiten detailliert erklärt.

5.4 Evaluation Hardware

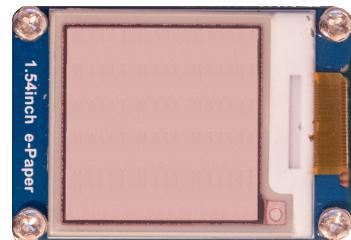
Auf Grundlage der bestehenden Prototyps soll neue Hardware evaluiert werden. Der Funktionsumfang des Prototyps soll gleichwertig bleiben. Sind die Komponenten erst definiert, wird das Material bestellt.

Für den Aufbau des zweiten Prototypen wurde die nachfolgende Hardware verwendet.

Gerätetyp	Model	Bild
Entwicklungsboard	Espressif ESP-WROOM-32 Core Dev Kit	
GPS Module	ublox Neo 6M	
Lora Modul	RFM96	
Speaker	Piezo Speaker	

Display

eink Display Waveshare
1.54 Zoll 200x 200



Schalter

gewöhnlicher 4 Pin Schalter



Aus Zeitgründen habe ich mich bei den Komponenten von bestehenden Projekten resp. Einträge im Internet inspirieren lassen. Es gab kein spezielles Auswahlverfahren für die restlichen Komponenten. Meiner Meinung nach nicht das beste Vorgehen. Bei einem erneutem Projekt in dieser Richtung müsste ich vorher die Kriterien für die Komponenten festlegen.

5.4.1 Lora-Antenne

Ich beging den Fehler, anzunehmen, dass beim Kauf des Lora-Moduls eine Antenne mitgeliefert wird. Da ich über sehr wenig Erfahrungen in der Telekommunikationswelt verfügte, wollte ich mich auch mit diesem Thema etwas genauer befassen. Ich entschied mich gegen eine erneute Bestellung und baute mir die Antenne einfach selbst. Nach einer kurzen Internetrecherche fand ich heraus wie mit einfachen Mitteln eine zuverlässige Antenne für 868Mhz gebaut werden kann [11]. Ein einfaches Kabel genügt. Das Kabel muss die Länge der Wellenlänge oder einen Bruchteil bspw. 1/2 oder 1/4 der Wellenlänge haben. Die Berechnung der Wellenlänge ist mit der nachfolgenden Formel möglich [12].

Es gilt $\lambda = \text{Wellenlänge}$, $v = \text{Übertragungsgeschwindigkeit (Lichtgeschwindigkeit)}$, $f = \text{Durchschnitt der Frequenz (Lora Frequenz in Europa)}$

$$\begin{aligned}\lambda &= \frac{v}{f} \\ \lambda &= \frac{299.792.458m/s}{868.000.000} = 34,54cm \\ \frac{\lambda}{2} &= \frac{34,53cm}{2} = 17,27cm \\ \frac{\lambda}{4} &= \frac{17,27cm}{2} = 8,63cm\end{aligned}$$

17,27cm sind für meinen Prototypen zu lange. Ich entschied mich 1/4 der Wellenlänge zu verwenden. Für eine Antenne musste also nur ein Kabel mit der Länge **8.63cm** vorbereitet und mit dem Antennenport des Lora Moduls verbunden werden.

Das Resultat
TODO BILD

Die Kommunikation funktioniert einwandfrei und ist mit der des ersten Prototypen vergleichbar. Mir ist bewusst, dass der Empfang mit einer professionellen Antenne noch besser werden würde. Die Thematik 'Antenne' müsste bei einer Weiterführung des Projektes nochmals beleuchtet werden.

5.5 Zusammenbau Hardware

Die Komponenten werden zusammengebaut und untereinander verbunden. Zur besseren Übersicht wird ein Schema der Elektronik erstellt.

5.5.1 Schema

Die Kommunikation unter den Komponenten wurde im Laufe der Projektarbeit für mich immer unübersichtlicher. Um mir die Arbeit zu erleichtern, habe ich ein Schema mit den Komponenten erstellt. Vor der Arbeit hatte ich noch nie ein Schema gezeichnet und musste mich zuerst einige Stunden in die Materie einarbeiten. Auf anraten einiger Kollegen, mit Erfahrung im Elektronik Umfeld, sowie der frei verfügbaren Lernressourcen und Vorlagen, fiel meine Wahl auf das Programm **Eagle** der Firma Autodesk [17]. Dank der guten Dokumentation und den Lernvideos auf diversen Plattformen bspw. Youtube kam ich beim Zeichnen schnell vorwärts.

Ich begann damit Vorlagen der Elektronischen Bausteine/Chips finden. Die Vorlagen zu Lora, GPS, ESP, Schalter und Piezo Lautsprecher konnte ich teilweise vom Hersteller oder von der Community herunterladen und in Eagle importieren. Für das Display musste ich eine bestehende Vorlage eines Displays anpassen. Ich habe dabei die Größe und die Pinbelegung verändert, damit das Element wie das verwendete Display (Waveshare eInk 200x200) aussieht.

Das Schema ist auf der nachfolgenden Seite abgebildet.

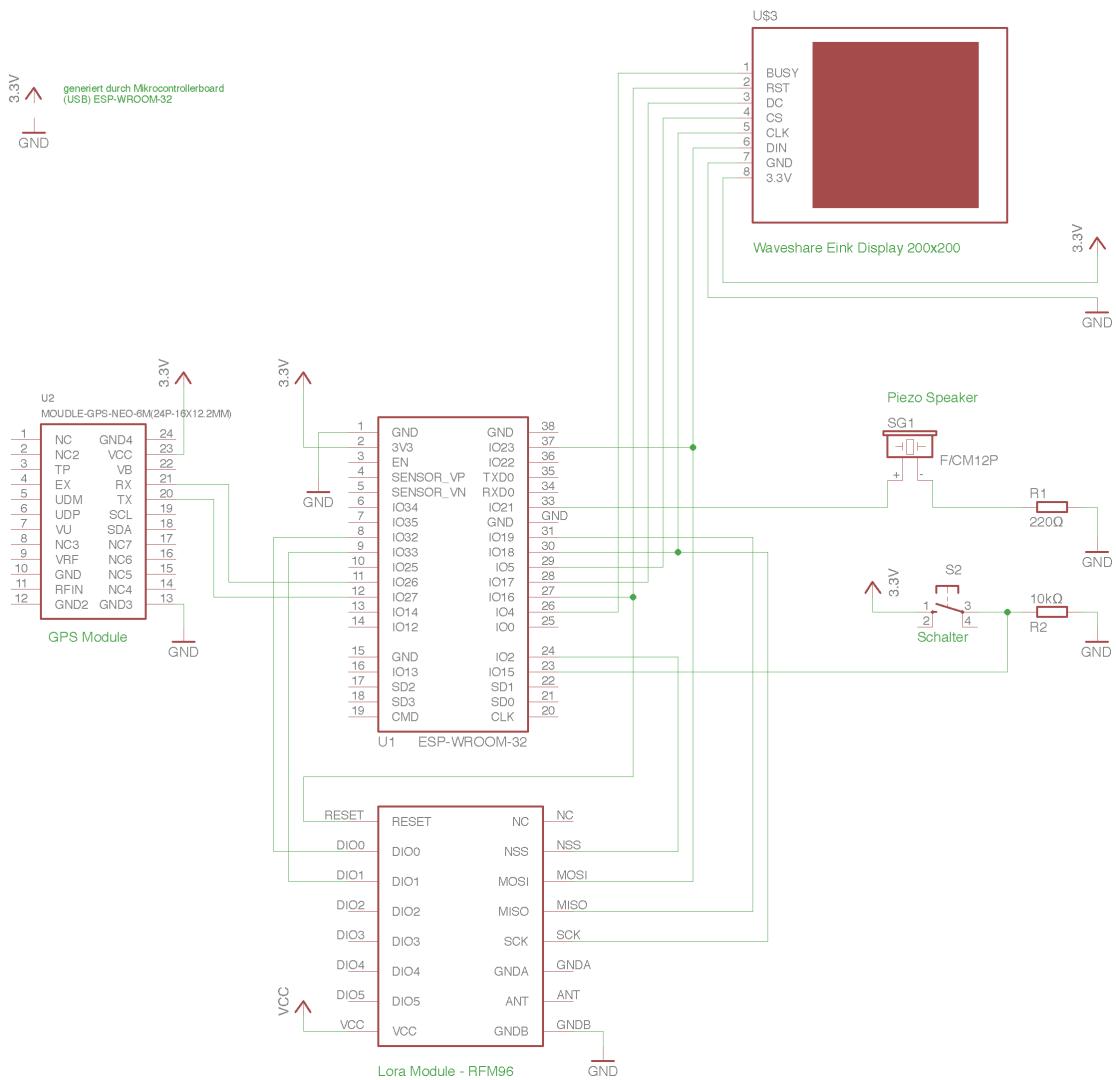


Abbildung 5.3: Prototyp 2 Schema

5.5.2 Mikrocontroller - Pinbelegung

Die nachfolgende Pinbelegung wurde verwendet um den ESP32 mit den übrigen Komponenten zu verbinden. Zu meiner Überraschung sind die Pins nicht fix durch den ESP32 festgelegt. Die Pins können meist für jegliche Funktion bspw. PWM, SPI verwendet werden. Durch diese Freiheit wird die Verkabelung massiv vereinfacht und die Anzahl der möglichen Fehlerquellen wird reduziert.

MCU Pin	Modul	Funktion
IO02	Lora	SPI ChipSelect (CS)
IO04	Lora + Display	SPI Busy
IO05	Display1	SPI ChipSelect(CS)
IO15	Switch	-
IO16	Lora + Display	SPI Reset
IO17	Display	DataCommand (DC)
IO18	Lora + Display	SPI Clock (CLK)
IO19	Lora + Display	Master In Slave Out (MISO)
IO21	Piezo Speaker	-
IO23	Lora + Display	SPI Master Out Slave In (MOSI)
IO26	GPS	UART RX
IO27	GPS	UART TX
IO32	Lora	DIO0
IO33	Lora	DIO1

5.6 Kommunikation

In der nachfolgenden Übersicht wird die Kommunikationsart zwischen den Komponenten aufgezeigt.

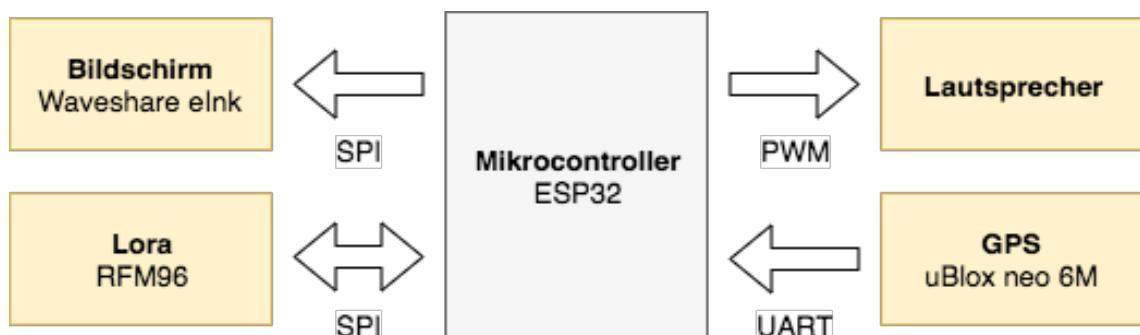


Abbildung 5.4: Kommunikationsart Prototyp 2

5.6.1 Zusammenbau

Die Komponenten wurden auf einer Entwicklungsplatine (PCB Breadboard) platziert und verlötet resp. angeschraubt. Der Mikrocontroller wurde absichtlich nicht im Zentrum platziert, damit der Zugang zum USB Anschluss frei blieb. Abschließend wurde gemäß Schema die Komponenten untereinander verkabelt.

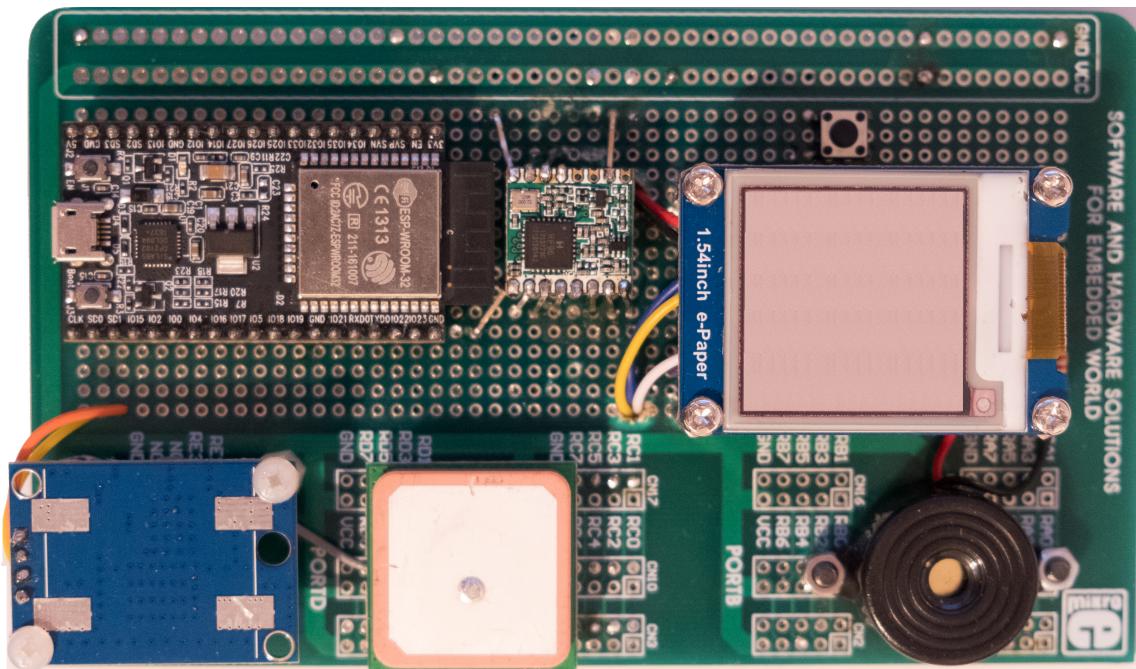


Abbildung 5.5: Prototyp 2

5.7 Software

Der bestehende Sourcecode des Prototyps 1 kann leider nicht eins zu eins übernommen werden. Die Software ist sehr stark für die Raspberry Pi Umgebung angepasst und ist daher nicht mit dem ESP32 Umfeld kompatibel. Die Software muss portiert werden. Die Logik der Software kann übernommen werden.

Pro Hardware Modul wird ein separates Software Modul erstellt.

5.7.1 Sourcecode

Hinweise wo Sie den Sourcecode einsehen können, finden Sie im Abschnitt 'Anhang'.

5.7.2 Entwicklungsumgebung

Durch den Umstieg vom Raspberry Pi auf den Mikrocontroller ESP32 hat sich im Bereich der Entwicklungsumgebung einiges verändert. Die ESP32 Plattform unterstützt, Stand heute (16.11.2017), zwei Entwicklungsumgebungen. Das wäre einerseits ESP-IDF (IoT Development Framework) sowie Arduino[18]. Die beiden Umgebungen sind miteinander kombinierbar. So kann in der IDF Umfeld die Arduino Unterstützung als Komponente importiert werden. Bestehende für Arduino konzipierte Bibliotheken können dadurch ebenfalls verwendet werden. Ich habe mich in Absprache meines Betreuers für die ESP-IDF Umgebung entschieden und mein System entsprechend vorbereitet. Die Installation der Entwicklungsumgebung ist nicht schwer, bedingt aber einige manuelle Schritte. Die Umgebung besteht aus einer Reihe von Tools (Toolchain) zum Kompilieren, Fehlersuche und dem Flashen des Mikrocontrollers.

Ich verzichte an dieser Stelle auf eine detaillierte Auflistung aller Installationsschritte. Die Installation wurde gemäss der Anleitung auf der Webseite der Firma Espressif durchgeführt [19].

Als Hostsystem wurde ein macOS System verwendet. Damit das System den ESP korrekt erkennen konnte, musste ein zusätzlicher Treiber installiert werden [20].

5.7.3 Probleme

Infolge der Umstellung auf die ESP32 Umgebung merkte ich relativ schnell, dass es keine extra für ESP32 portierte LMIC Library gab. Für die LoRaWan Kommunikation wurde daher auf eine bestehende Arduino Library (Arduino-LMIC) zurückgegriffen [1]. Den Kommunikationsablauf selbstständig zu programmieren hätte den Umfang dieser Arbeit gesprengt.

Leider schien die LMIC Library nicht zu 100% kompatibel. Nach einigen Websuchen fand ich heraus, dass auch andere User mit Problemen kämpfen. Scheinbar gibt es ein Problem beim Timing der Prozesse im Zusammenspiel mit RTOS.

Thread

Samuel and rmh78

Samuel Dec 29th at 11:52 AM
in #lmic

I am currently working on a ESP32 field test device, and I am thinking about moving the os runloop to a task in the second core, so the timing does not depend on the main program

1 reply

 **rmh78** 4 days ago
I have tried to run LMIC in a task and ran into massive timing issues. I think there is no current LMIC hal implementation for the esp32 right now. It would be nice if someone could port LMIC to esp32 with RTOS.

Abbildung 5.6: LMIC Problem

Es kam leider immer wieder vor, dass der Tracker einige für Ihn bestimmte Downlink Nachrichten nicht empfangen konnte. Diese Nachrichten enthalten die Angaben zu Gefahrenzonen uns sind daher sehr wichtig.

Das Problem besteht bis zum heute (7.1.2018).

5.7.4 Display

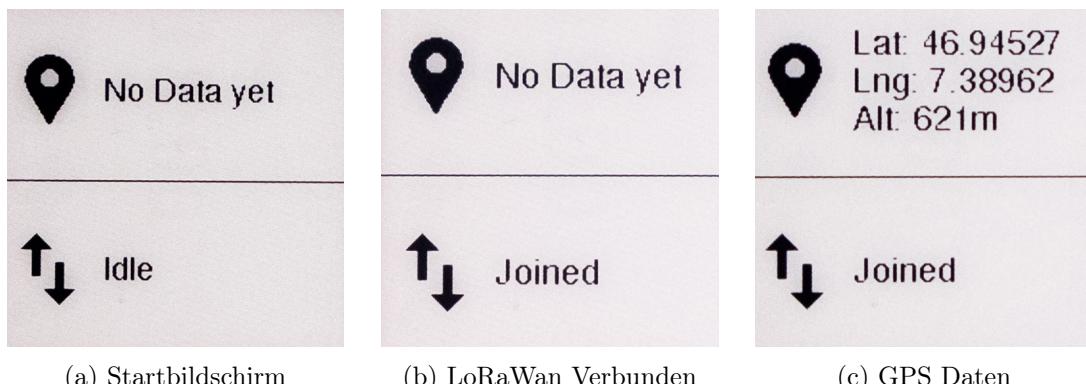
Um die Alarmierung des Alpinisten zu verbessern wurde beim Tracker ein Display eingebaut. Um die Batterie des Trackers nicht weiter zu belasten wurde ein eInk Display ausgewählt. Das Display benötigt nur Energie wenn das angezeigte Bild ändert.

Das Display hat drei Status.

Dashboard

Das Dashboard gibt Hinweise zur Lokalisierung (GPS) und der Datenkommunikation (LoRaWan). Das Dashboard ist die Standardansicht, wenn keine Gefahr droht. Im oberen Teil des Bildes werden die GPS Koordinaten und Höhe über Meer angezeigt. Im Unteren Teil finden sich Informationen zur Lora Kommunikation dh. der Verbindungsstand und der Zeitpunkt der letzten Übertragung (Uplink). Diese Informationen sind für den Alpinisten ein Indikator ob das Gerät einwandfrei funktioniert.

Ein Auszug möglicher Darstellungen auf dem Bildschirm:



Alarm

Der Alarm Bildschirm informiert den Alpinisten über mögliche Gefahren.



(a) Lawine (b) Steinschlag (c) Schneesturm (d) Spalte

Notfall

Der Notfall Modus wird vom Alpinisten selbst ausgelöst. Der Bildschirm dient der Bestätigung für das erfolgreiche aktivieren des Alarms.



Abbildung 5.9: Notfallalarm

5.7.5 Klassendiagramm

Die einzelnen physikalischen Bausteine (GPS, Lora...) werden in Software Klassen abgebildet. Die Klassen besitzen entsprechende Methoden zum auslesen resp. steuern der Komponenten. Um die Software zu visualisieren wurde ein Klassendiagramm erstellt.

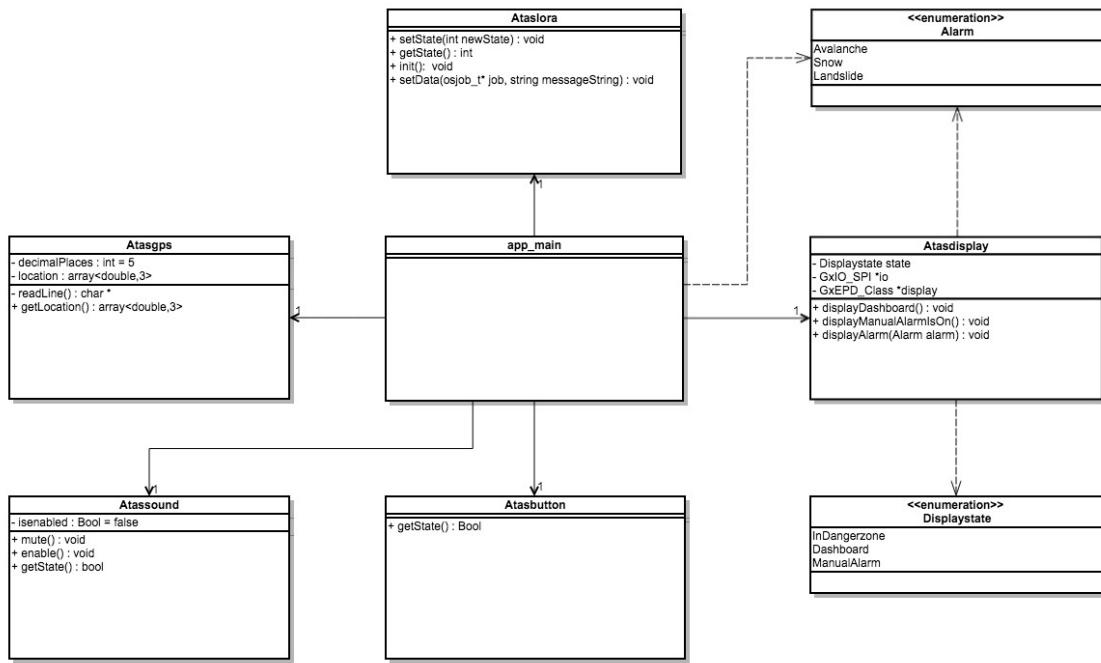


Abbildung 5.10: Klassendiagramm

5.8 Finalisierung & Testing

Abschließend werden einige Tests mit dem neuen Tracker durchgeführt.

5.8.1 Energieversorgung

Einer der Hauptgründe für den Bau eines neuen Prototypen ist der Energieverbrauch. In einem abschließenden Test soll nun festgestellt werden, wie sich der Energiebedarf des ersten zum zweiten Prototypen verändert hat. Folgende Szenarien sollen geprüft werden.

Ohne Daten zu senden

Während dem Senden von Daten

TODO, PRIO 1

Kapitel 6

Testing

6.1 Ablauf Vorgehen

Ein Teil des Systemaufbaus aus dem vorgängigen Projekt soll getestet werden. In diesem Kapitel wird genau beschrieben, wie beim Testing des Systems vorgegangen wird.

In der nächsten Abbildung sind die geplanten Schritte aufgeführt.

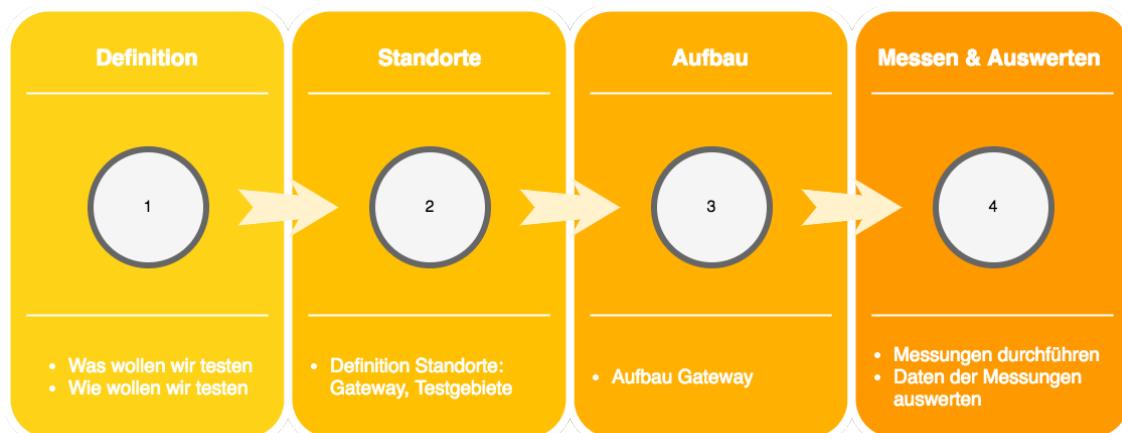


Abbildung 6.1: Ablauf Testing

Die einzelnen Schritte werden auf den kommenden Seiten detailliert erklärt.

6.2 Definition

Es muss definiert werden

- welche Komponenten geprüft werden sollen

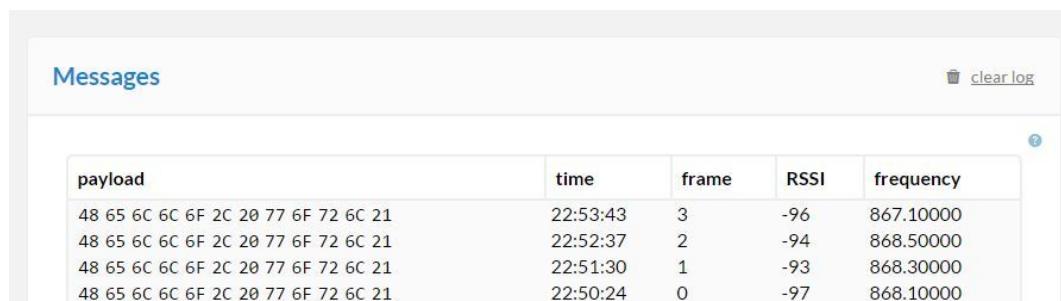
Während der Arbeit soll die Kommunikation zwischen Atas-Node und Atas-Gateway getestet werden.

- welche Aspekte der Komponente sollen betrachtet werden

Der Fokus liegt auf der **Zuverlässigkeit** der Übertragung. Mich interessiert, wie sicher es ist, dass eine Meldung ihr Ziel erreicht.

- mit welchen Mitteln resp. Messinstrumenten gemessen wird

Das TTN Netzwerk bietet uns die Möglichkeit mittels Dashboard einzusehen, wann welche Pakete auf dem Gateway eintreffen. So können wir prüfen ob die Verbindung zwischen Gateway und Tracker funktioniert.



The screenshot shows a table titled "Messages" with the following data:

payload	time	frame	RSSI	frequency
48 65 6C 6C 6F 2C 20 77 6F 72 6C 21	22:53:43	3	-96	867.10000
48 65 6C 6C 6F 2C 20 77 6F 72 6C 21	22:52:37	2	-94	868.50000
48 65 6C 6C 6F 2C 20 77 6F 72 6C 21	22:51:30	1	-93	868.30000
48 65 6C 6C 6F 2C 20 77 6F 72 6C 21	22:50:24	0	-97	868.10000

Abbildung 6.2: TTN Messages

- welche Daten können untersucht werden

Für die Messungen sind mehrere Daten von Interesse. Dazu gehören

- Signalstärke
- Wie viele Pakete gehen verloren
- Distanz der Übertragung
- Sichtverhältnisse zwischen Sender und Empfänger

6.2.1 Schlussfolgerung

Abgeleitet aus der vorhergehenden Definitionen ergibt sich die folgende Aufgabenstellung im Bereich Testing. Es muss eine Testumgebung aufgebaut und die Verbindung getestet werden. Um von den bestehenden TTN Gateways unabhängig zu sein, wird ein **eigener Gateway** aufgebaut. Anschließend werden mehrere Tests durchgeführt. Die Verbindung wird aufgebaut und mittels TTN Dashboard geprüft.

6.3 Standort

Ausgehend von den vorangegangen Definitionen muss folgendes definiert werden.

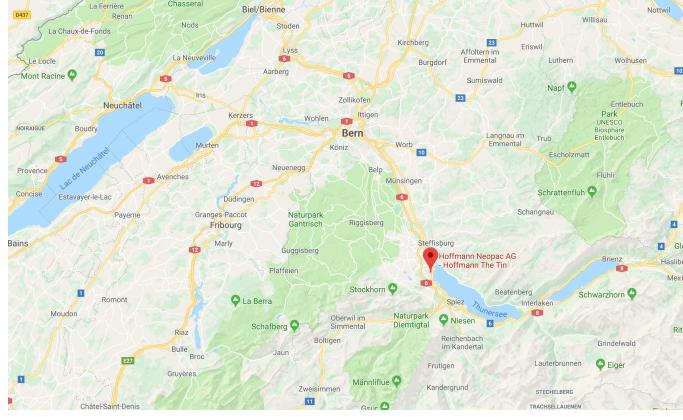
- Es muss ein Ort gefunden werden, wo der Gateway platziert werden kann.

Der Standort kann frei gewählt werden. Zwei Kriterien müssen zwingend erfüllt sein.

1. Der Gateway empfängt die Daten der Endgeräte und muss diese dem TTN Netzwerk über das Internet weiterleiten können. Eine stabile Internetverbindung ist nötig.
 2. Der Standort sollte in der Nähe von möglichen Wanderrouten resp. Testbereichen liegen.
- Für die Tests sollten mind. 3 Routen/Wanderungen definiert werden. Dies garantiert eine vernünftige Anzahl an Messwerten.
 - Auf dem Pfad selbst werden Orte definiert wo die Verbindungstests durchgeführt werden.

6.3.1 Gateway

Der Standort für den LoRa Gateway war schnell gefunden. Ich arbeite momentan als Informatik Systemengineer bei der Hoffmann Neopad AG. Die Firma besitzt eine Niederlassung in der Nähe der Stadt Thun (BE). Das Gebäude ist 2 Stockwerke hoch und die Adresse liegt in unmittelbarer Nähe der Voralpen. Hohe Berge für einen möglichen Testlauf sind max. 20km entfernt. Dazu zählen beispielsweise das Stockhorn, Niesen sowie das Niederhorn.

Beschreibung	Bild
Standort Hoffmann Neopac AG	 A detailed map of the Bernese Oberland region in Switzerland, centered around the city of Bern. A red dot marks the location of Hoffmann Neopac AG in Thun. The map shows various towns, lakes (e.g., Biel/Bienne, Neuchâtel), and mountains. Major roads and railway lines are also visible.
Standort Gateway auf Gebäude	 A photograph of a modern, two-story office building with a glass facade and a dark frame. An arrow points from the top left towards the entrance area. The building is surrounded by a paved driveway and some greenery. Other industrial buildings are visible in the background under a clear blue sky.

6.3.2 Testregionen

Im ersten Teil meiner Arbeit konzentrierte ich mich stark auf den Aufbau des Prototypen. Das Testing hatte für mich eher eine kleinere Priorität. Ein Fehler wie sich später herausstellen sollte. Der Winter ließ, wie sonst nicht üblich, nicht lange auf sich warten. Bereits Mitte Oktober beginn es zu schneien. Wanderrouten/ Regionen welche ich für das Testing verwenden wollte, bspw. der Nordaufstieg Stockhorn, wurden wegen zu hoher Lawinengefahr geschlossen.

Nichtsdestotrotz konnte ich 2 Testregionen definieren, welche einfach zugänglich und soweit gefahrlos passierbar waren.

Region 1: Niederhorn	Bild
Region	
Höhenunterschied zum Gateway	
Mittlere Distanz zum Gateway	Xm

Region 2: Heiligenschwendi

Bild

Region



Höhenunterschied zum Gateway



Mittlere Distanz zum Gateway

7Km

6.4 Aufbau & Vorbereitungen

6.4.1 Gateway

In diesem Abschnitt ist der Aufbau des LoRaWan Gateways beschrieben. Der Gateway besteht aus den folgenden Komponenten

- A: Raspberry Pi 3
- B: IMST iC880a Board [8]
- C: IP67-Box

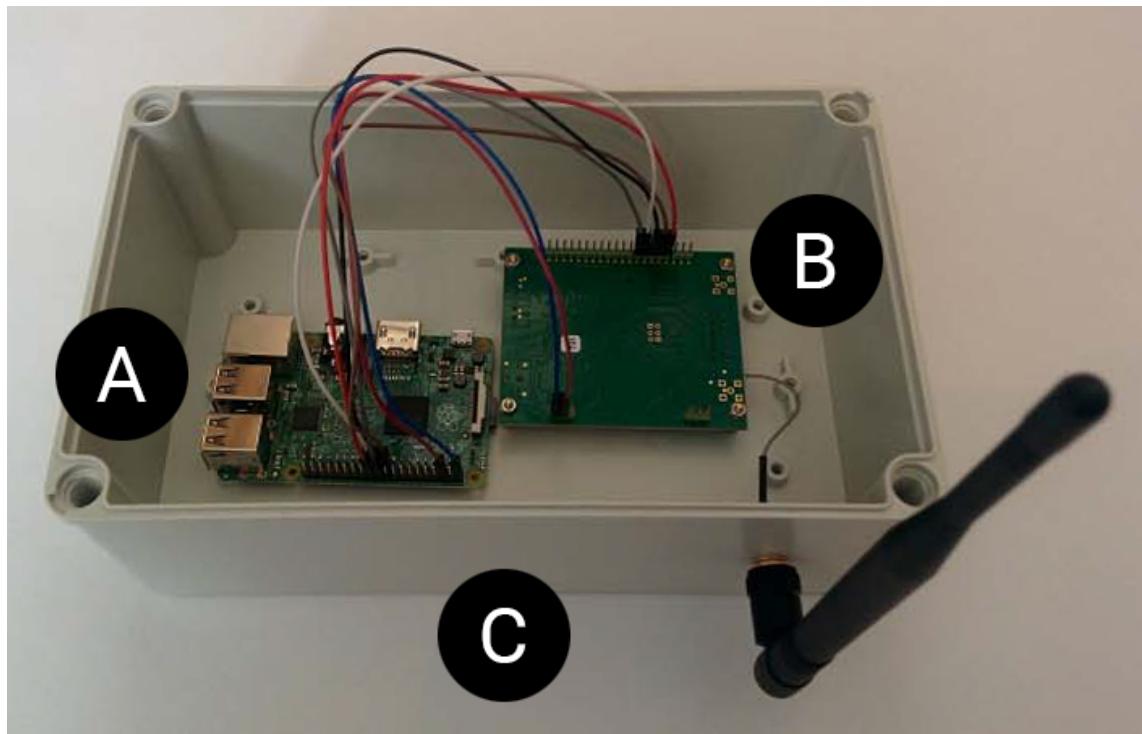


Abbildung 6.3: Gateway

Durch den Einsatz einer IP67 Box sind die elektronischen Komponenten gegen Staub und Wasser geschützt. Leider habe ich auch hier zu wenig Zeit in die Planung investiert. Wie sollte ich den Pi mit Strom versorgen wenn die Box geschlossen ist? Ein zusätzliches Loch musste gebohrt werden um das Kabel verlegen zu können.

Installation

Die Webressourcen der TTN Community von Zürich waren bei der Installation eine große Hilfe [7]. Das Raspberry PI und das iC880 Board wurden gemäß der Webseite verkabelt.

Die Community hat ein sehr simples Installationsskript zusammengestellt um die Gateway Software automatisiert zu installieren. Folgende Schritte wurden auf dem Raspber PI durchgeführt.

```
// update software
sudo apt-get update
sudo apt-get upgrade
sudo apt-get install git

// set local & timezone
sudo dpkg-reconfigure locales // de_CH
sudo dpkg-reconfigure tzdata // Europe/Zurich

// Enable SPI, Interfacing options -> P4 SPI
sudo raspi-config

// Reboot
reboot

// Install Gateway Software
git clone https://github.com/ttn-zh/ic880a-gateway.git ~/ic880a-gateway
cd ~/ic880a-gateway
sudo ./install.sh spi
```

Alles in allem eine sehr einfache Prozedur die nicht viel zeit in Anspruch genommen hat. Anhand der Mac Adresse des Raspberry PI wird im Abschluss des Skripts eine EUI generiert.

TTN Registrierung

Mit der vorher generierten EUI kann der Gateway nun auf dem TTN Dashboard registriert werden.

REGISTER GATEWAY

Gateway EUI
The EUI of the gateway as read from the LoRa module
`EB 82 7E BF FF E9 B8 00` 8 bytes

I'm using the legacy packet forwarder
Select this if you are using the legacy [Semtech packet forwarder](#).

Description
A human-readable description of the gateway
Atas Gateway Thun Gwatt

Frequency Plan
The [frequency plan](#) this gateway will use
Europe 868MHz

Router
The router this gateway will connect to. To reduce latency, pick a router that is in a region which is close to the location of the gateway.
ttn-router-eu

Location
The exact location of your gateway. This will be used if your gateway cannot determine its location by itself. Set a location by clicking on the map.


Abbildung 6.4: TTN Dashboard Gateway

Der Gateway ist nun online und leitet Datenpakete weiter.

GATEWAYS

Gateway ID	Status	Region	
eui-b827ebfff9b800e	atas-gateway01	connected	EU_863_870

Abbildung 6.5: TTN Dashboard Gateway Registriert

6.5 Messungen

Es werden verschiedene Testszenarien definiert.

6.5.1 Test 1: Praxistest

Vorgehen

Das ATAS System soll geprüft werden ob es praxistauglich ist. Um dies zu überprüfen werden in den vorher definierten Testregionen Messungen durchgeführt. Es soll getestet werden, wie zuverlässig die Übertragung funktioniert.

Für diesen Test werden die folgenden Daten erhoben

1. Distanz zwischen Sender und Empfänger
2. Signal to Noise Ratio (SNR)
3. RSSI
4. Sichtverhältnisse zwischen Sender und Empfänger

Die erhobenen Daten werden nach Sichtverhältnissen kategorisiert. Messungen sollen in 3 Kategorien eingeteilt werden.

1. A: Direkter Sichtkontakt
2. B: Wenig Sichtkontakt resp. teilweise blockiert. Bspw. durch Bewaldung oder Steinformationen
3. C: kein ein Sichtkontakt zwischen Empfänger und Sender.

Die Tests werden mit einem fixen Einstellungen durchgeführt

1. Spreadingfactor SF12
2. Bandbreite 125KHz
3. Coding Rate 4/5

Resultate

TODO, PRIO 1

6.5.2 Test 2: Spreadingfactor

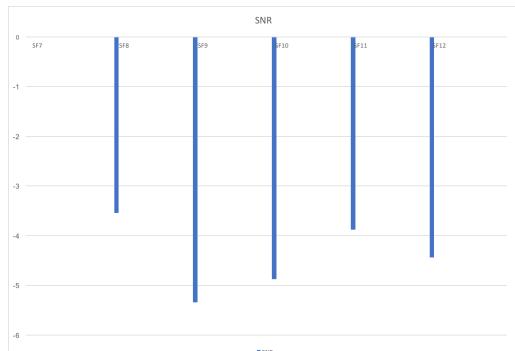
Es soll geprüft werden, wie sich der Spreadingfactor resp. Datenrate auf die Übertragung auswirkt. Für diesen Test werden die folgenden Daten erhoben

1. Spreading Factor
2. Signal to Noise Ratio
3. RSSI

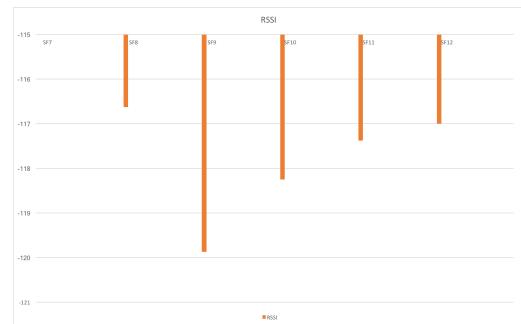
Die Messungen werden in der Region 1 durchgeführt. Der Standort wird nicht gewechselt. Die Distanz zwischen Gateway und Tracker bleiben also gleich. Es werden pro Spreading Factor (7-12) 10 Messungen durchgeführt. Pro 10 Messwerte wurden jeweils der mindeste wie auch der maximale Messwert d.H. Ausreißer entfernt.

Gemäß Literatur müsste klar erkennbar werden, dass die SNR Werte und die RSSI Werte mit einem höheren SF besser werden, d.H das Signal stärker beim Gateway eintrifft.

Die Messresultate:



(a) Messwerte SNR pro Spreadingfactor



(b) Messwerte RSSI pro Spreadingfactor

Wie bei den Resultaten ersichtlich, ist die Annahme nicht eingetroffen. Es ist kein Muster erkennbar, das bei einem höheren SF ein besseres Signal zu erwarten ist. Ich gehe davon aus, dass für einen solchen Test mehr Messwerte von Nöten sind. Ich werde den Test also mit mehr Messwerten wiederholen müssen. Für SF7 konnten keine Pakete zugestellt werden. Entweder hatte der Gateway eine Fehlfunktion, oder die Distanz war zu Gross als dass dieser die Daten hätte empfangen können.

6.5.3 Test 3: Isolation Schnee

Kapitel 7

Schlusswort

7.1 Verbesserungen bestehendes System

TODO, Prio 2

7.2 Ausblick ATAS

TODO, Prio 2

7.3 Rückblick Projekt

TODO, Prio 3

Selbständigkeitserklärung

Ich bestätige, dass ich die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der im Literaturverzeichnis angegebenen Quellen und Hilfsmittel angefertigt habe. Sämtliche Textstellen, die nicht von mir stammen, sind als Zitate gekennzeichnet und mit dem genauen Hinweis auf ihre Herkunft versehen.

Ort, Datum: Bern, 15.01.2018

Namen Vornamen: Martin Schmidli

Unterschriften:

Anhang

7.4 Software

Es wird kein Code direkt an dieses Dokument angehängt. Zur Verwaltung des Sourcecodes wurde die Plattform Github verwendet. Jegliche Commits vor dem 16.09.2017 gehören zur Projekt 2 Arbeit. Commits nach diesem Datum wurden im Zuge der Bachelor Thesis erstellt. Folgend Sie den Links um den Sourcecode einzusehen.

7.4.1 Atas-Webapp

<https://github.com/schmm2/atas-webapp>

7.4.2 Atas-Service

<https://github.com/schmm2/atas-service>

7.4.3 Atas-Node

Software welche auf dem 1. Prototypen eingesetzt wird

<https://github.com/schmm2/atas-node>

7.4.4 Atas-Node2

Software welche auf dem 2. Prototypen eingesetzt wird

<https://github.com/ATAS-Group/atas-node2>

7.5 Libraries

Für den Aufbau der ATAS Umgebung wurde auf diverse Softwarebibliotheken zurückgegriffen. Da ich sehr dankbar für dessen Bereitstellung bin, möchte ich diese hier namentlich erwähnen.

7.5.1 Atas-Node2

Bibliothek	Verwendungszweck	Url
LMIC-Arduino	LoRaWan Kommunikation	https://github.com/ matthijskooijman/ arduino-lmic

Abbildungsverzeichnis

1	Motivation Alpen Panorama	4
1.1	Bergunfallstatistik 2016 - Notfallsituationen nach Ursachen	7
2.1	Atas Schema Grob	11
3.1	Systemübersicht Detail	15
3.2	Tracker Detail	16
3.3	Gateway Detail	17
3.4	TTN Detail	18
3.5	MQTT Struktur	19
3.6	TTN Detail	20
4.1	OTAA Ablauf	26
4.2	LoRaWan Class A	28
4.3	LoRaWan Class B	28
4.4	LoRaWan Class C	29
5.1	Prototyping Ablauf	34
5.2	Flowchart Prototyp 2	35
5.3	Prototyp2 Schema	40
5.4	Kommunikationsart Prototyp 2	41
5.5	Prototyp 2	42
5.6	LMIC Problem	44
5.9	Notfallalarm	46
5.10	Klassendiagramm	47
6.1	Flowchart Testing	49
6.2	TTN Messages	50
6.3	Gateway	55
6.4	TTN Dashboard Gateway	57
6.5	TTN Dashboard Gateway Registriert	57

Glossar

ATAS Aplinist Tracking & Alerting System. 2

Literaturverzeichnis

- [1] <https://github.com/matthijskooijman/arduino-lmic> 2.2.2018
- [2] <https://www.thethingsnetwork.org/article/setting-up-a-private-handler-connected-to-the-public-community-network> 2.2.2018
- [3] <https://www.thethTNNingsnetwork.org/docs/applications/apis.html> 2.2.2018
- [4] <https://www.rs-online.com/designspark/learning-lorawan-basics> 1.2.2018
- [5] <https://www.thethingsnetwork.org/wiki/LoRaWAN/ADR> 1.2.2018
- [6] <http://www.erodocdb.dk/Docs/doc98/official/pdf/REC7003e.pdf> 1.2.2018
- [7] <https://github.com/ttn-zh/ic880a-gateway/wiki> 1.2.2018
- [8] <https://wireless-solutions.de/products/radiomodules/ic880a> 1.2.2018
- [9] <https://www.bergwelten.com/a/die-schoensten-zitate-rund-ums-bergsteigen> 1.2.2018
- [10] <http://www.sac-cas.ch/unterwegs/sicherheit/bergnotfallstatistik.html> 1.2.2018
- [11] <https://www.thethingsnetwork.org/forum/t/antenna-length-for-868-and-433-mhz/5378> 2.1.2018
- [12] <https://en.m.wikipedia.org/wiki/Wavelength> 2.1.2018
- [13] http://www.oegan.at/notfallmedizin/index.php?option=com_content&view=article&id=76:unterkuehlung-als-ueberlebenschance&catid=42&Itemid=118 1.01.2018
- [14] <https://www.thethingsnetwork.org/wiki/LoRaWAN/Security> 27.07.2017
- [15] <https://www.jaguar-network.com/en/news/lorawan-in-a-nutshell-2-internet-of-things-iot>, 27.07.2017
- [16] <https://www.thethingsnetwork.org/wiki/LoRaWAN/ADR> 27.07.2017

- [17] <https://www.autodesk.com/products/eagle> 27.07.2017
- [18] <http://iot-bits.com/documentation/esp32-tutorial-and-example-programs/>
20.12.2017
- [19] <https://dl.espressif.com/doc/esp-idf/latest/get-started/index.html>
27.12.2017
- [20] <https://www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers> 27.12.2017