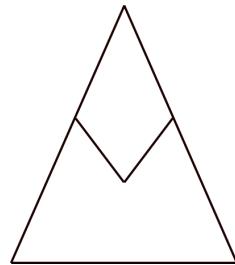




BERN UNIVERSITY OF APPLIED SCIENCES

BACHELOR THESIS
STUDIENGANG INFORMATIK

Alpinist Tracking & Alerting System



Autor:
Martin SCHMIDLI

Betreuer:
Mohamed MOKDAD

Experte:
Daniel VOISARD, BAKOM

Bern, 16. Januar 2018

Inhaltsverzeichnis

1 Einleitung	5
1.1 Projektidee	7
1.2 Vorarbeiten	7
1.3 Aufgabenstellung	7
1.4 Rahmenbedingungen	7
1.5 Technologien	8
1.6 Anwendungsfälle	8
2 Systembeschreibung - Grob	9
2.1 Grobe Systemarchitektur	9
2.2 Aufbau	10
3 Evaluation Technologie	13
3.1 Wireless Datenkommunikation	13
3.2 LoRa	16
3.3 LoRaWan	22
3.4 LoRaWan Netzwerk	29
4 Systembeschreibung - Detail	32
4.1 Überblick	33
4.2 Tracker	34
4.3 Gateway	36
4.4 Datenhandler	37
4.5 Logik	39
4.6 Webapplikation	40
4.7 Kommunikation	44
5 Prototyp	45
5.1 Begründung neuer Prototyp	45
5.2 Ablauf Prototyping	46
5.3 Ablauf Vorgehen	47
5.4 Evaluation Hardware	48
5.5 Zusammenbau Hardware	52

5.6	Kommunikation	55
5.7	Software	57
5.8	Finalisierung & Testing	62
6	Testing	66
6.1	Ablauf Vorgehen	66
6.2	Definition	67
6.3	Standort	69
6.4	Aufbau & Vorbereitungen	73
6.5	Messungen	78
7	Schlusswort	85
7.1	Verbesserungen bestehendes System	85
7.2	Ausblick	85
7.3	Rückblick Projekt	85
7.4	Software	87
7.5	Libraries	88

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 [17]. 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 5 Bereiche. Der Bericht startet mit einer groben Systembeschreibung und wird im Verlauf der Thesis detaillierter. Im ersten Teil der Thesis wird die grundsätzliche Funktionalität und der grobe Aufbau des Systems definiert. Im zweiten Teil geht es um die Technologien welche für das System verwendet werden sollen. Warum wurde diese gewählt und wie werden diese eingesetzt. Im dritten Teil wird der finale Aufbau definiert. Im Vordergrund stehen hier die Aufgabe der einzelnen Komponenten (Hard und Software) und deren Kommunikation untereinander. Im vierten Teil wird der Aufbau eines neuen Hardware Prototypen beschrieben. Der im Vorprojekt aufgebaute Prototyp weist einige Schwachstellen auf und musste verbessert werden. Im fünften Schritt wird das ATAS System getestet. Im Fokus liegen hier 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 grandios.



Abbildung 1: Motivation Alpen Panorama

Ich bin kein Extrembergsteiger. 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? Tausende von Fragen ergaben sich.

Diese Thematik beschäftigte mich noch lange. Als es 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.

Kapitel 1

Einleitung

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

Hans lander, Italienischer Bergsteiger[16]

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 oder Abstürze der Bergsteigenden. Der Schweizer Alpenclub (SAC) führt dazu eine jährliche Statistik [17]. Nachfolgend ein kleiner Auszug aus der Bergunfallstatistik 2016. Die Grafik zeigt eine Übersicht der Anzahl Situationen bei welchen eine Rettungsdienst ausrücken musste. Zu den Rettungsdiensten zählt beispielsweise die REGA.

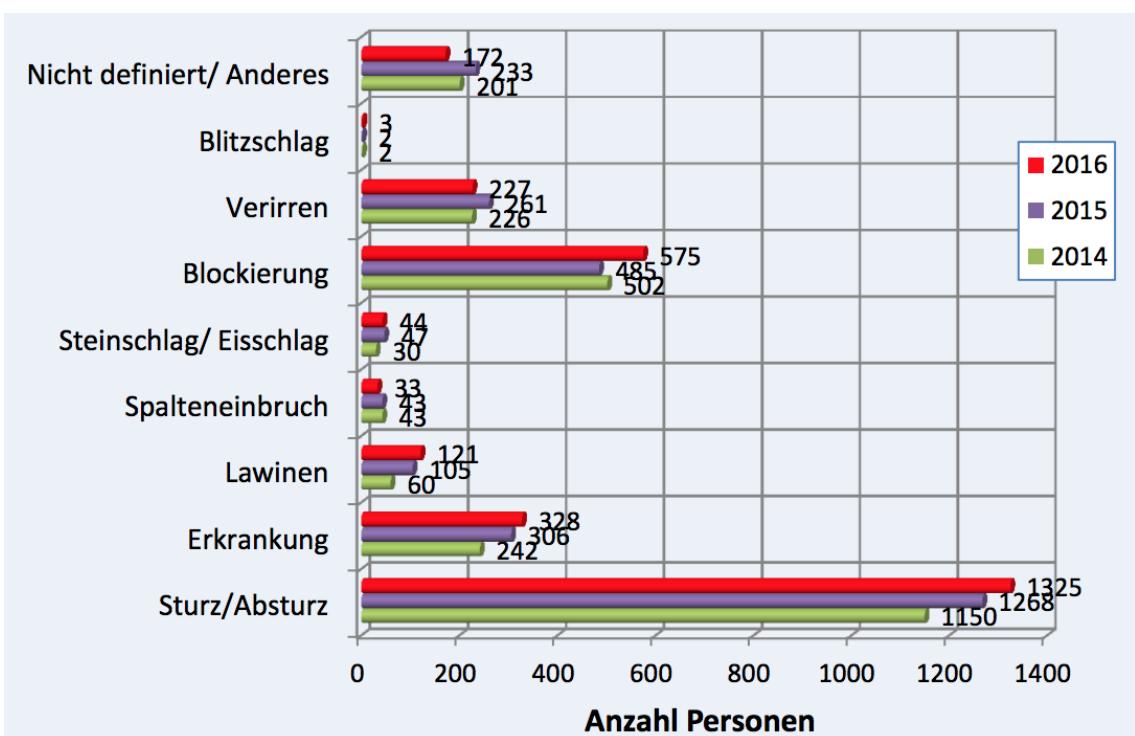


Abbildung 1.1: Bergunfallstatistik 2016 - Notfallsituationen nach Ursachen

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

Ein nicht geringer Anteil machen Unfälle mit Lawinen aus. Wird eine Person innert 15 Minuten geborgen Leben noch 90% der Betroffenen. Man spricht auch von der Überlebensphase. Nach 15 Minuten folgt die Erstickungsphase, die Überlebenschancen schwinden auf geringe 30%. Je schneller ein Rettungsdienst alarmiert werden kann desto höher sind die Überlebenschancen [1].

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. Der Empfänger wird bei der Talstation oder im nächsten Bergdorf montiert. Die Daten werden von einer Software gesammelt und analysiert. 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 Wanderung. 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

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.
- 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, TTN 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.

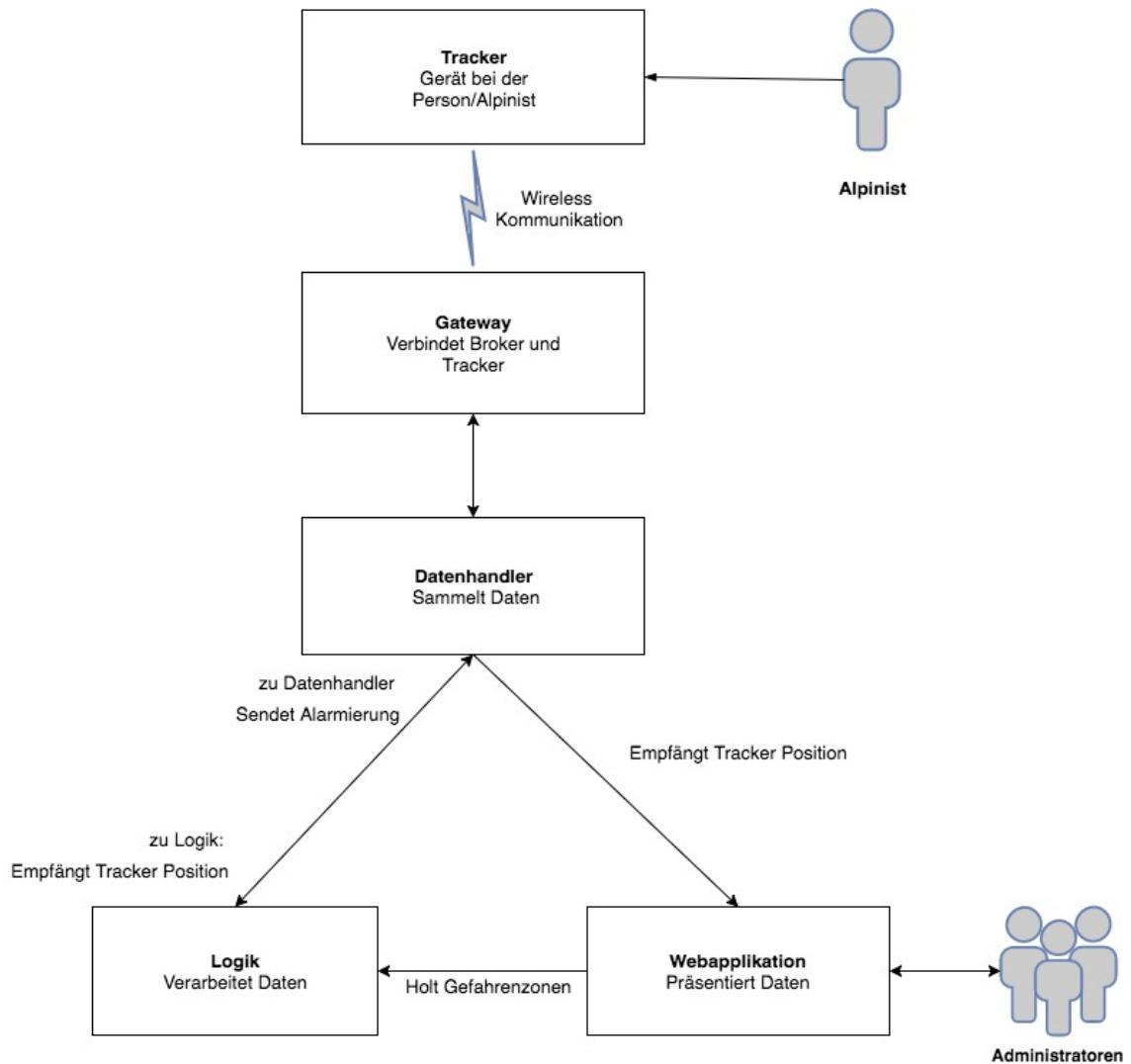


Abbildung 2.1: Atas Schema Grob

2.2.2 Benutzer

Die nachfolgenden Gruppen wurden als Benutzer des Systems identifiziert.

Alpinist

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 aufgelisteten Komponenten. Komponenten sind als Hardware oder Software zu verstehen.

Tracker

Stellen Sie sich ein kleines mobiles Gerät vor. Nachfolgend nennen wir dieses Gerät Tracker. Der Alpinist trägt den Tracker bei sich bspw. in einem Rucksack. Der Tracker zeichnet auf wo sich die Person momentan befindet

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

Der Tracker überträgt folgende Daten an den Gateway.

- Notrufsignale
- GPS Position des Trackers

Befindet sich der Tracker resp. die Person in einer Gefahrenzone wird die Person über den Bildschirm und mit einem Tonsignal alarmiert.

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 **Datenhandler** genannt.

Datenhandler

Sammelt und speichert Daten. Der Datenhandler kommuniziert mit den Gateways. Zusätzlich bietet der Datenhandler ein Interface zum Abfragen und senden von Daten an die Tracker via Gateway.

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.

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 Datenhandler 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

Evaluation Technologie

In diesem Kapitel wird genauer auf die eingesetzten Technologien eingegangen. Es wird beschrieben wofür und weshalb diese Technologien eingesetzt werden.

3.1 Wireless Datenkommunikation

Zu Beginn dieses Projektes musste festgelegt werden, welche Technologie für die Datenübertragung verwendet werden soll. Nachfolgend werden drei mögliche Technologien analysiert.

3.1.1 Sigfox

Sigfox ist eine LPWAN Lösung welche vom Netzwerk Design eher an ein herkömmliches Mobilfunknetz erinnert. Sigfox ist eher ein geschlossenes System. Wurde das Gebiet mit Sigfox erschlossen, kann das Netzwerk gegen Bezahlung mitbenutzt werden. Der Aufbau eines eigenen Antenne scheint nicht möglich.

Vorteile

- Durch den geringen Energiebedarf halten batteriebetriebene Geräte länger
- Große Reichweite in nicht bebauten Gebieten, bis zu 40km

Nachteile

- Zum jetzigen Zeitpunkt Stand 15.1.2018 wurde die Schweiz von Sigfox noch nicht erschlossen.
- Geringe Bandbreite, ca. 100B/s
- Limitierung in der Zahl der Nachrichten die gesendet werden können. Zum Endgerät (Downlink) 140x 12 Byte. Vom Endgerät an das Netzwerk (Uplink) 4x 8 Byte
- Sigfox wird als Service Angeboten. Für den Dienst muss bezahlt werden.

3.1.2 LoRa

LoRA steht für Long Range und ist eine Wide-Area Netzwerk Lösung. Die Lösung besteht meist aus zwei Schichten

- LoRA. Link Layer, Physikalische Schicht. LoRa definiert, wie die Daten im Medium (Luft) übertragen werden.
- LoRAWan. Media Access Control (MAC) Protokoll. Definiert das Format der Daten.

LoRa hat drei Hauptmerkmale. Der Energieverbrauch ist gering (low-power), die Reichweite ist hoch (long-range) und die maximale Übertragungsrate ist gering (low-throughput).

Vorteile

- Durch den geringen Energiebedarf halten batteriebetriebene Geräte länger
- Die Kosten für die Installation eines Lora resp. LoRaWan Gateways sind gering +- 300.-
- Große Reichweite in nicht bebauten Gebieten, bis zu 15km
- Geringe Störanfälligkeit durch den Einsatz von Spreading Factors resp. Chirp Spread Spectrum (CSS)
- Frequenzbereich im ISM Band. Lizenzfreie Frequenzbereiche. Es müssen keine Gebühren bezahlt werden.
- Großes Link Budget, +14dB Sender und -137db Empfänger Sensitivität → 151dB
- Erfreut sich grosser Beliebtheit bei der Community. Das verwenden von LoRa gestaltet sich sehr einfach. Immer mehr Informationen und hilfreiche Ressourcen sind im Internet zu finden.

Nachteile

- Geringe Verbreitung je nach Region
- Geringe Bandbreite, von 0.3Kb/s zu 27Kb/s mit CSS, 50Kb/s mit FSK Modulation [7]

3.1.3 NB-IOT, LTE Cat M1

NB-IOT (Narrowband - Internet of Things) und LTE Cat M1 wurden als Konkurrenz zu den populären Lösungen LoRa und Sigfox entwickelt. Auf diese Netzwerke wird hier vorerst nicht eingegangen. Die Standards wurden erst kürzlich von der Industrie aufgegriffen und werden langsam eingeführt. Die Swisscom plant erste Tests gegen Ende 2017. Der Rollout für die Nutzer soll im Frühjahr 2018 beginnen [2].

3.1.4 Entscheid

Sigfox scheidet als mögliche Technologie aus. Die Angaben zur Reichweite sind zwar sehr beeindruckend, die Beschränkung in der Anzahl der Down und Uplinks macht es leider für das ATAS System unbrauchbar. Mit nur 140 Nachrichten pro Tag, kann die Position des Trackers nicht häufig genug ermittelt werden.

NB-IOT & LTE Cat M1 scheinen eine potente Mobilfunklösung für den IoT Bereich zu sein. Die Zukunft wird zeigen ob sich die Technologie gegen die bestehende Konkurrenz behaupten kann.

Bleibt noch LoRa. LoRa überzeugt mit der guten Reichweite und der kleinen Belastung der Batterie. Für das System ATAS wird LoRa als Datenkommunikationslösung eingesetzt.

3.2 LoRa

Um das Projekt durchzuführen, musste das Wissen zum Thema LoRa vertieft werden.

3.2.1 LoRa Modulation

LoRa unterstützt zwei Modulationsarten, das simple Frequency Shift Key (FSK) und das Chirp Spread Spectrum (CSS) Verfahren. In dieser Arbeit setzen wir den Fokus auf CSS.

Ein Chirp ist ein sinusähnliches Signal, bei welchem über die Zeit die Frequenz erhöht oder reduziert wird.

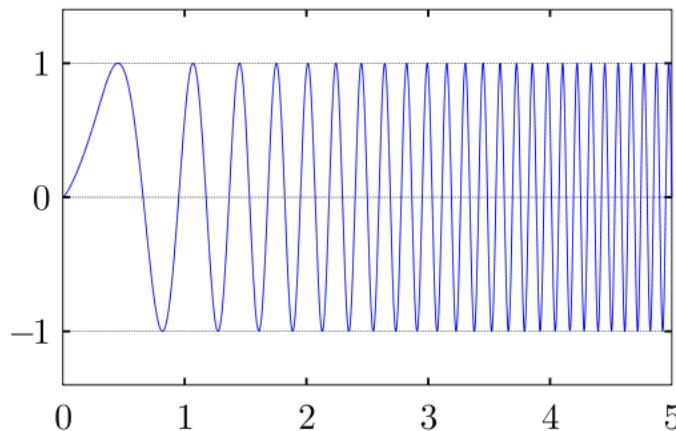


Abbildung 3.1: Beispiel Upchirp

Wie im Bild ersichtlich erhöht sich die Frequenz des Signals. Bei einer Erhöhung der Frequenz reden wir von einem Upchirp, bei einer Reduktion von einem Downchirp. Das Chirp Signal wird anschließend verwendet um die Nutzdaten zu kodieren. Pro übertragenes Symbol wird ein Chirp generiert.

Bandbreite

Bei LoRa haben wir die Möglichkeit zwischen drei möglichen Bandbreiten zu wählen. Die Bandbreite definiert die größe des Bereiches der zwischen der kleinste und der größten möglichen Frequenz. Die Bandbreiten sind 125kHz, 250kHz und 500kHz. Eine Verdopplung der Bandbreite (BW125 zu BW250) erlaubt es doppelt soviele Bytes zu senden. Nur die Bandbreiten 125KHz und 250KHz sind in Europa zulässig.

Spreading Factor

Zwei Parameter beeinflussen das Chirp Moduliertes Signal. Erstens die Bandbreite und zweitens die sog. Sweep Rate.

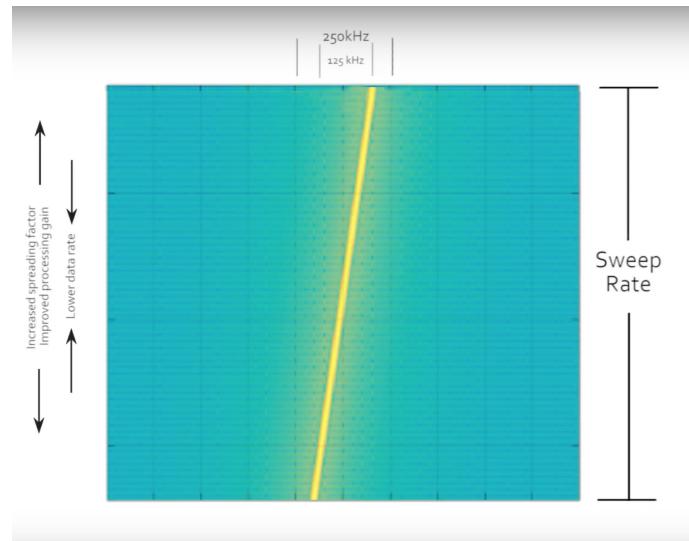


Abbildung 3.2: Wasserfall Darstellung: Chirp Parameter

Ein weiteres Beispiel eines Upchirps. Wir sehen in der Mitte die mögliche Bandbreite 125KHz und 250kHz. Die Frequenz nimmt gegen Links zu. Bei diesem Signal wird 125kHz verwendet. Der gelbe Strich zeigt uns die momentane Frequenz zum Zeitpunkt x. Nach erreichen der maximalen Frequenz folgt der nächste Chirp. Dieses Intervall wird auch Sweep Rate genannt.

Die Einstellungsmöglichkeiten der Parameter wurden in Europa zu Gruppen, sogenannten Spreading Factors zusammengefasst. Es gibt die Faktoren SF7-SF12.

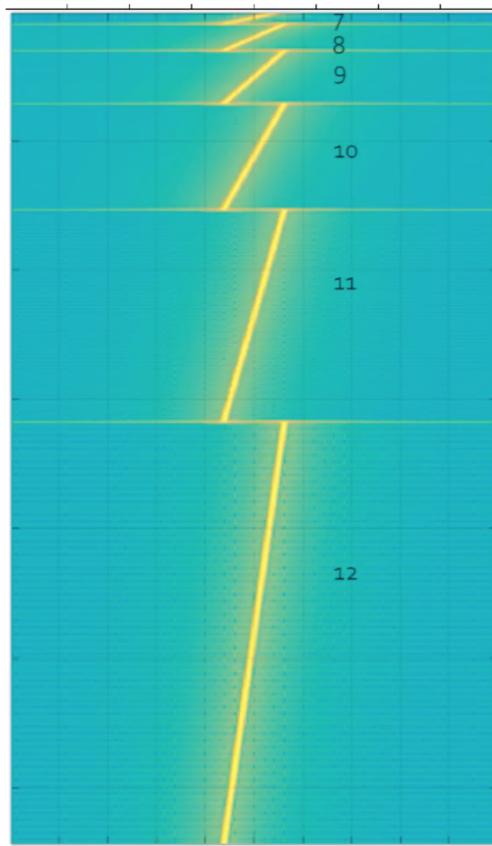


Abbildung 3.3: Wasserfall Darstellung: Spreadingfactors 7-12

Wie in der Grafik ersichtlich ist die Sweep Rate bei SF12 viel langsamer als bei SF7. Pro Schritt zwischen den SF (SF7 zu SF8) halbiert sich der Datendurchsatz. Das Senden einer Nachricht dauert länger ist aber einfacher zu dekodieren. Das Signal wird weniger anfällig gegenüber Störungen und kann damit über eine größere Distanz übertragen werden. Müssen 'größere' Datenmengen oder in einem kleineren Zeitintervall versendet werden, bietet es sich an einen kleineren Spreadingfactor zu wählen (7-9). Sollen große Distanzen überwunden werden, oder gibt es in der Region viele Störsignale, sollte ein hoher Spreadingfactor (10-12) verwendet werden.

3.2.2 Regulationen

Für die Schweiz und in vielen Teilen Europas gilt für die Nutzung von LoRa die Regelung ERC/REC 70-03 [13]. Zusammengestellt wurden die Regulationen von der CEPT, der European Conference of Postal and Telecommunications. Die Schweiz ist Mitglied dieser Organisation. Die Standardisierung selbst hat die Organisation ETSI (EN300.220) übernommen. Das Dokument beschreibt die verwendbaren Frequenzen, die maximale Übertragungsleistung (dbm) sowie den Duty Cycle.

Die Regulationen wurden in dieser Projektarbeit eingehalten. Die maximale Sendeleistung von 25mW resp. 14dbm wurde während den Tests nicht überschritten.

3.2.3 Datenübertragungsrate

Der Duty Cycle und die damit verbundene Airtime haben einen grossen Einfluss auf die maximale Datenübertragungsrate.

Duty Cycle

Der Duty Cycle oder zu Deutsch 'Auslastungsgrad' beschreibt, wie lange ein Signal oder ein System aktiv ist [4]. In unserem System beispielsweise wie lange der Tracker sendet. Dieser Prozentsatz wird meist vom Gesetzgeber festgelegt. Wir müssen uns an die Einschränkungen halten, was wiederum bedeutet, dass wir nicht konstant Daten senden dürfen. Die nachfolgenden Frequenzbänder & Duty Cycles wurden durch die ETSI definiert und können in Europa verwendet werden[?].

Band	Frequenz	Energieverbrauch	Duty Cycle
g	863.0 - 868.0 MHz	25mW	1%
g1	868.0 - 868.6 MHz	25mW	1%
g2	868.7 -869.2 MHz	25 mW	0.1%
g3	869.4 - 869.65 MHz	500mW	10%
g4	869.7 - 870.0 MHz	5mW/25mW	1%

Diese Frequenzbänder gelten für den Einsatz in Kombination mit LoRaWan in Europa. In anderen Regionen der Welt gelten andere Frequenzen.

Airtime

Die Airtime definiert, wie lange es dauert ein Signal vom LoRa Endgerät zum Gateway zu übertragen. Die Airtime ist abhängig von der Menge der zu übertragenden Daten und dem verwendeten Spreading Factor.

Die untenstehende Grafik verbildlicht diesen Zusammenhang

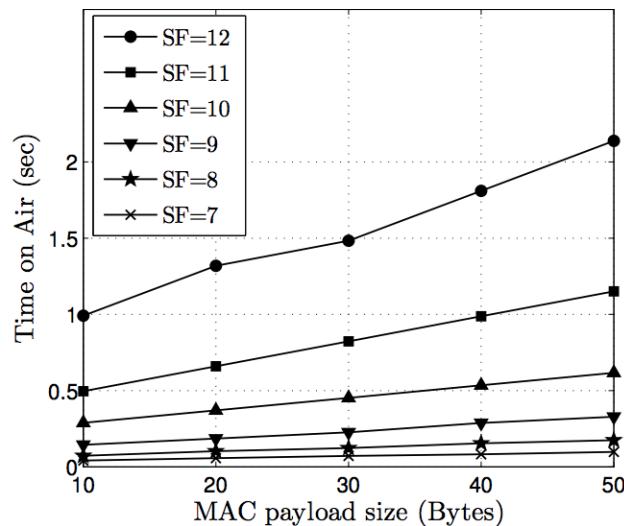


Abbildung 3.4: Time on Air

Je höher der Spreadingfactor, desto länger dauert die Übertragung. Auf die genaue Berechnung der Airtime und der gesamten Payload wird hier nicht weiter eingegangen.

Berechnung

Wir wissen nun also, dass wir nicht beliebig oft senden dürfen. Weiter wissen wir, dass die Airtime im Endeffekt bestimmt wie lange die Übertragung dauert. Die Formel lautet nun wie folgt.

$$time = \frac{airtime}{duty cycle * 1000}$$

Im Abschnitt 'Testing' werden wir später genau ermitteln wie häufig wir mit dem ATAS System senden können.

3.2.4 Coding Rate

Die Coding Rate definiert die Error Kodierungsrate. Eine höhere Kodierungsrate ermöglicht es das Signal auch bei Störungen (Noise) zu rekonstruieren. Ein erhöhen der Coding Rate verursacht eine größere Menge an Daten die gesendet werden → Overhead. Nutzdaten (bit) werden mehrmals übertragen. Ein Signal beinhaltet also redundante Informationen. Mögliche Coding Rate Werte im LoRa Umfeld sind 4/5 und 4/8.

Beispiel 3.2.4.1. Annahme: Wir setzen die Coding Rate auf 4/8
 $8 - 4 = 4$, d.H bei 4 Bit Nützlichen Informationen fügen wir nochmals 4 Bit redundante Informationen hinzu.

Um ein möglichst gutes Signal zu erhalten, soll im ATAS System die Kodierungsrate auf 4/8 festgelegt werden.

3.3 LoRaWan

LoRaWan ist ein Protokoll welches auf LoRa aufbaut. Man kann es sich als das TCP/IP von LoRa vorstellen. Das Protokoll definiert die Struktur der Daten die gesendet werden.

3.3.1 Netzwerk Architektur

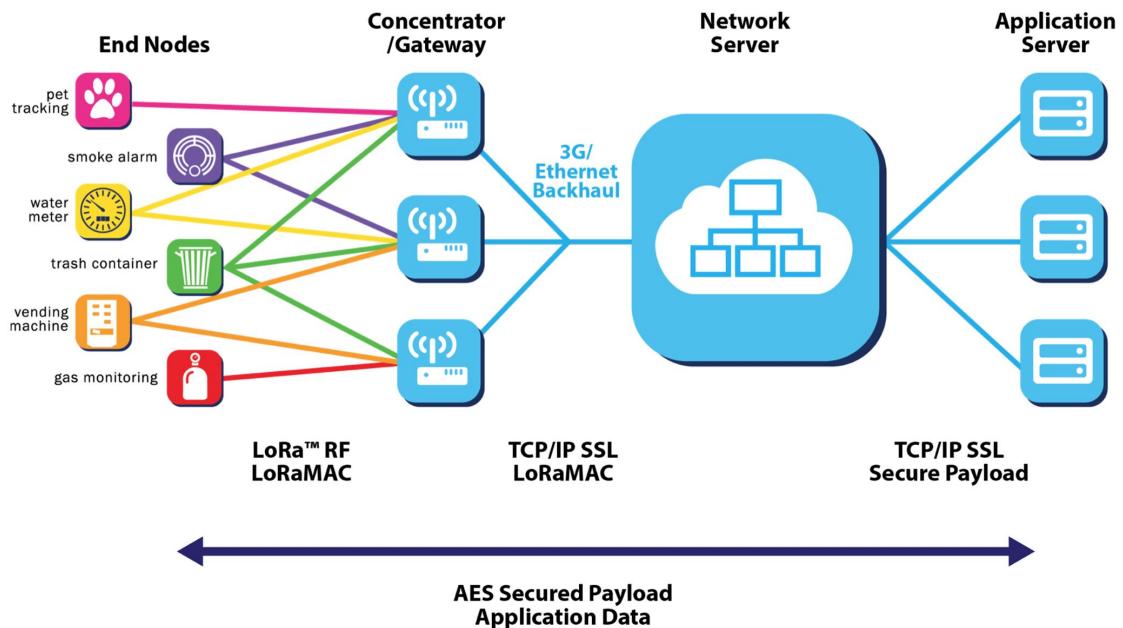


Abbildung 3.5: LoRaWan Architektur

Ein LoRaWan Netzwerk besteht aus mehreren Komponenten

Die End Nodes sammeln Daten über Sensoren. Das können ganz unterschiedliche Sachen sein, wie zum Beispiel Feuchtigkeit in einem Raum. Die End Nodes senden Daten an die Empfänger (Gateways, Concentrator) via LoRa. Die Daten werden hier, nicht wie in einem Mobilfunknetz an nur eine Antenne, sondern an alle Empfänger gesendet.

Die Gateways leiten die erhaltenen Informationen weiter an den Netzwerk Server. Die Gateway sind meist mit dem Internet über Kabel oder Wireless verbunden.

Der Netzwerk Server verwaltet die ein kommenden Datenströme. Doppelte Pakete werden gefiltert und Sicherheitsregeln werden angewandt. Abschliessend sendet der Gateway eine Empfangsbestätigung (Acknowledgement) an den Gateway, dass er die Pakete sauber erhalten hat.

Zum Schluss werden die Daten an die zuständigen **Applikations Server** weitergeleitet.

3.3.2 Sicherheit

Durch bereits eingebaute Sicherheitsmechanismen in LoRaWan sind die übertragenen Daten geschützt. Die Daten werden End zu End mit AES-128Bit verschlüsselt.

3.3.3 Endgerät Verbindungsmethoden

OTAA (Over-The-Air-Activation) und ABP (Activation By Personalization) sind Verbindungsmethoden für LoRaWan Endgerät in einem LoraWan Netzwerk.[22]. 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 stehlen, 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 angenommen 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 [22].

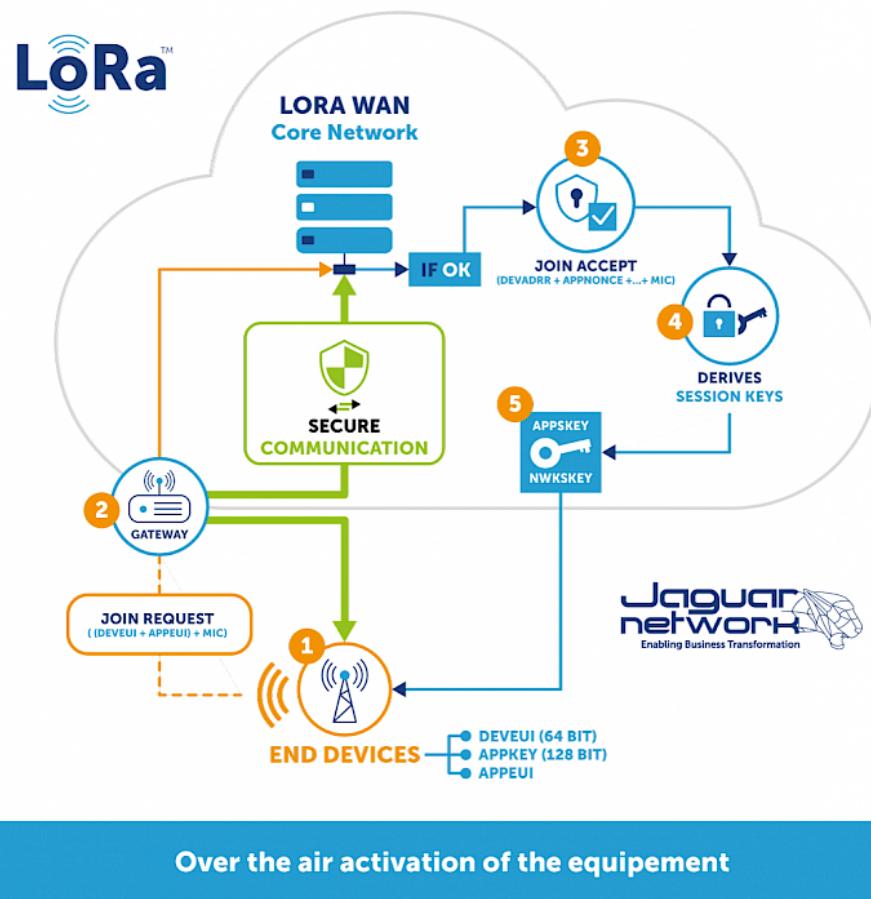


Abbildung 3.6: OTAA Ablauf

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 wurde während dem Projekt die Verbindungsmethodik von ABP auf OTAA umgestellt. Damit konnte die Sicherheit des Systems verbessert werden.

3.3.4 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.

3.3.5 Geräteklassen

LoRaWan Endgeräte werden in drei Geräteklassen unterteilt [11]. 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.

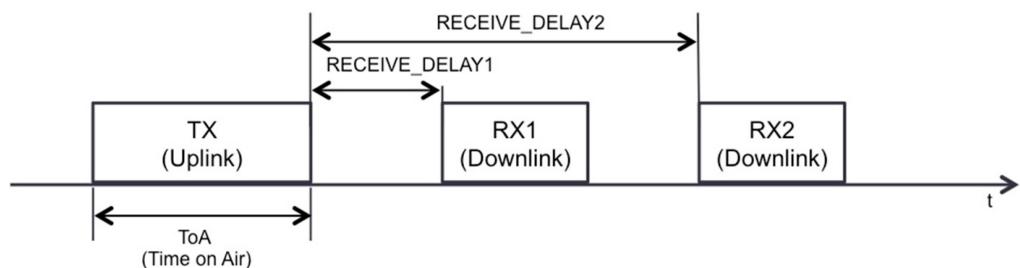


Abbildung 3.7: LoRaWan Klasse A

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.

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.

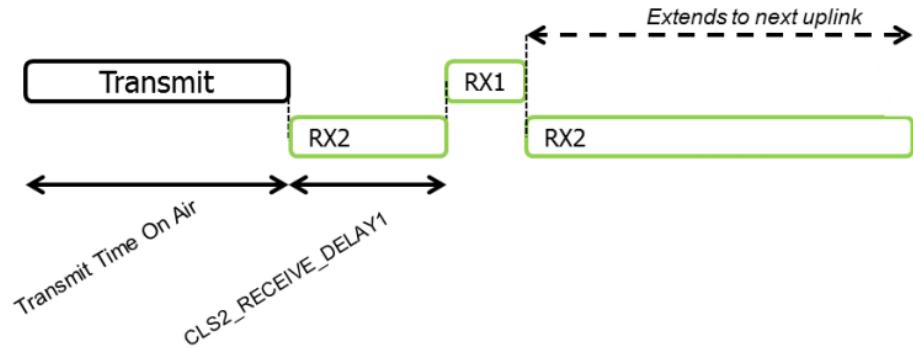


Abbildung 3.8: LoRaWan Klasse C

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.

3.3.6 Adaptive Data Rate (ADR)

Ist Adaptive Data Rate aktiv (ADR), entscheidet das LoraWan Endgerät selbständig ob und wie die Kommunikation optimiert werden soll. Gemäss TTN [23] 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 [12]. 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 einer hohen Signaldämpfung. Die nun gesendeten Uplinks erreichen möglicherweise das Ziel nicht.

Die Entwicklung eines eigenen ADR Systems wäre hier die Lösung. 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.

3.4 LoRaWan Netzwerk

3.4.1 Evaluation

Ein eigenes LoRaWan Netzwerk aufzubauen ist möglich, würde aber den Umfang dieser Arbeit sprengen. Aus diesem Grund wurde entschieden auf bereits bestehende Netzwerke zurückzugreifen.

Die Verbreitung von LoRaWan in der Schweiz ist mittlerweile sehr hoch. Unternehmen wie die Swisscom und die Schweizer Post haben begonnen große LoRaWan Netzwerke aufzubauen [5].

Eine Übersicht über die aktuelle Abdeckung des Swisscom , Stand (2.11.2017) [6].

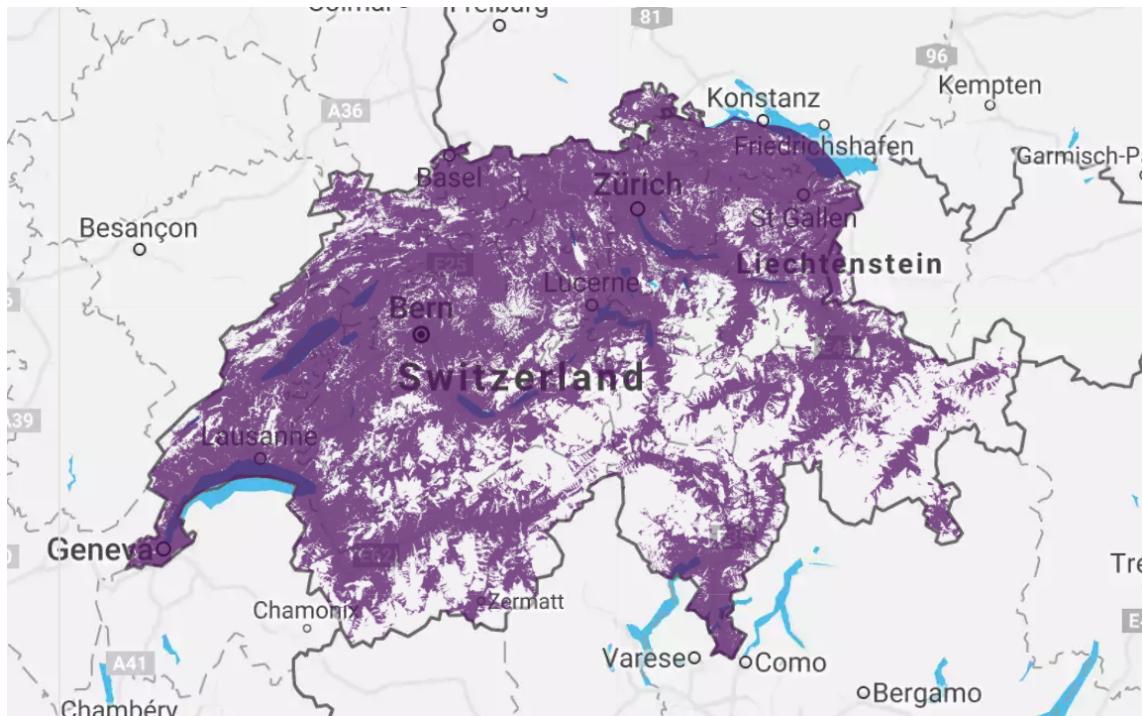


Abbildung 3.9: Swisscom LoRaWan Abdeckung

Vielott steht für Abdeckung. Weisse Stellen bedeutet es existiert keine LoRaWan Abdeckung. Wie in der Grafik ersichtlich, sind die Städtischen Gebiete bereits erschlossen. Anders sieht es in den Bergregionen aus. Das Netzwerk der Swisscom /Post kann somit für das ATAS System nicht verwendet werden.

3.4.2 The Things Network, TTN

Auf Anraten von Personen aus dem nahen Umfeld welche sich bereits intensiv mit LoRaWan auseinander gesetzt haben, wird beim ATAS System das Netzwerk 'The Things Network' eingesetzt. The Things Network kurz TTN ist eine LoRaWan Plattform, welche LoRaWan Endgeräte mit Applikationen verbindet. Der Beitritt ist kostenlos. Das Projekt wird von einer grossen Community betrieben. Ein eigener Gateway kann unkompliziert aufgeschaltet und in das Netzwerk integriert werden.

Architektur

Das TTN Netzwerk in sich besteht aus vielen einzelnen Komponenten.

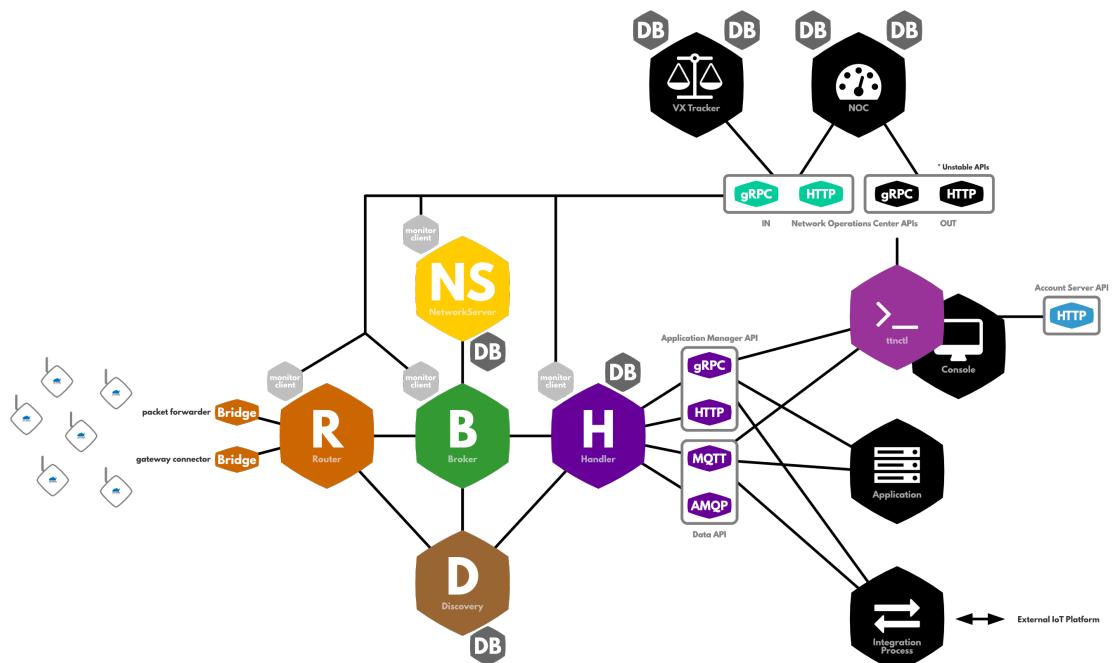


Abbildung 3.10: TTN Architektur

In dieser Arbeit wird **nicht auf die Details des TTN Netzwerkes eingegangen**. Wie das System in sich funktioniert hat keinen Einfluss auf das ATAS Projekt. Das TTN Netzwerk wird als Blackbox betrachtet. Die untenstehende Grafik verbildlicht diesen Ansatz. Über das TTN Netzwerk, via Gateways (G), werden Daten an die Tracker (N) gesendet. Die Tracker senden Ihre Daten via Gateway an das TTN. Das TTN Netzwerk wiederum bietet Schnittstellen an (API) um Daten zu senden und zu empfangen.

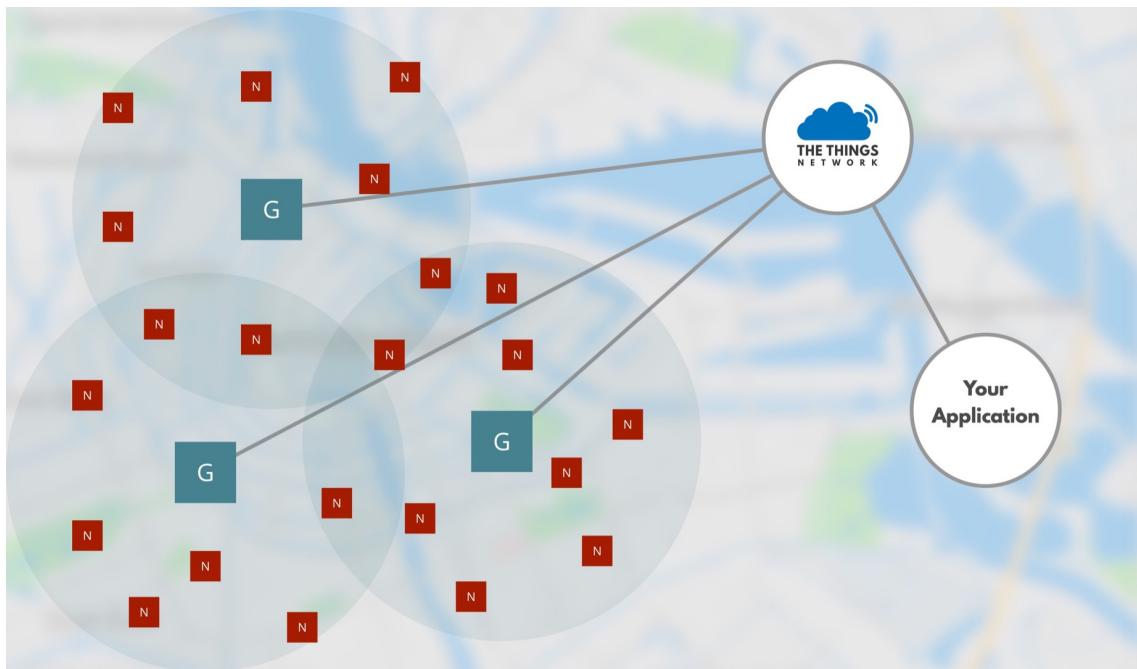


Abbildung 3.11: TTN Blackbox

3.4.3 Datenschnittstelle / API

Durch die Wahl von TTN als LoRaWan Netzwerk muss die Datenübertragungsart MQTT genutzt. Der Zugriff auf die Datenschnittstelle (API) von TTN ist nur per MQTT möglich. Die Unterstützung von AMQP ist geplant aber noch nicht verfügbar [10]. Das Protokoll MQTT war dem Projektteam bereits bekannt. Während dem Studium konnten bereits positive Erfahrungen mit MQTT gesammelt werden.

Kapitel 4

Systembeschreibung - Detail

Die vorhergegangene Architekturbeschreibung gibt einen grobe ersten Überblick über das System. Technische Details zu den verwendeten Technologien oder den genauen Fluss der Kommunikation waren 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 Box wurden im Rahmen dieser Arbeit selbständig erarbeitet. Um einen möglichst guten und kompletten Überblick über das System zu liefern, wird nicht zwischen Hard und Software unterschieden. Gelbe Wolken markieren einen Internetservice welcher für den Aufbau des Systems integriert wurde. Blaue Zylinder markieren eine Datenbanklösung. Das Abbild einer Person markiert einen User des Systems.



Abbildung 4.1: Legende - Diagram

Die im Absatz beschriebene Komponente wird als graue Box dargestellt und so hervorgehoben.

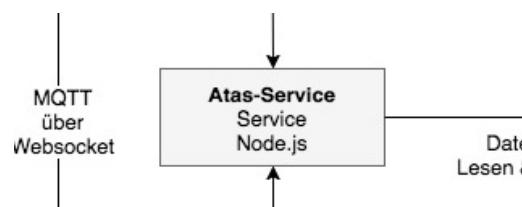


Abbildung 4.2: Legende - Diagram - graue Box

4.1 Überblick

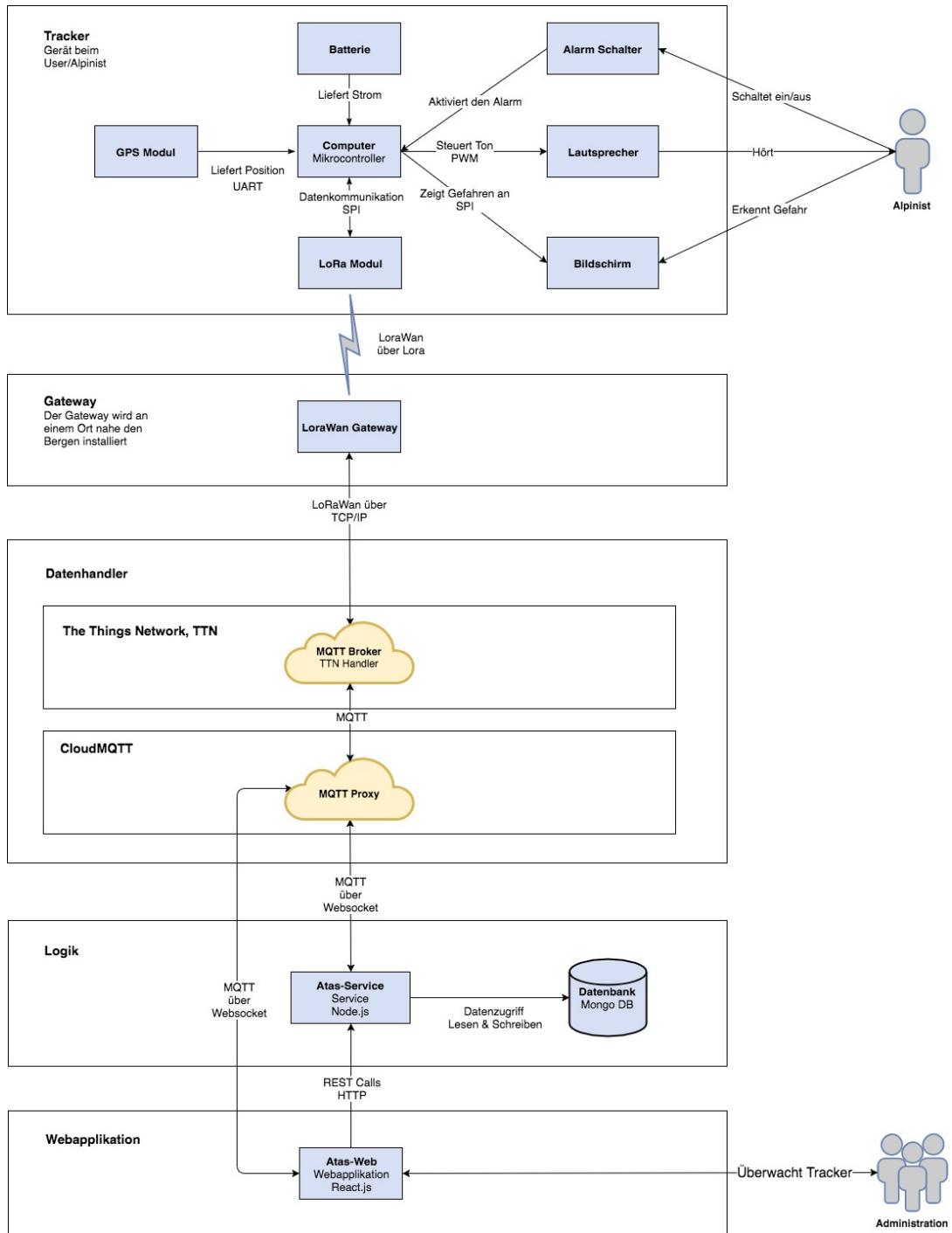


Abbildung 4.3: Systemübersicht Detail

4.2 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.

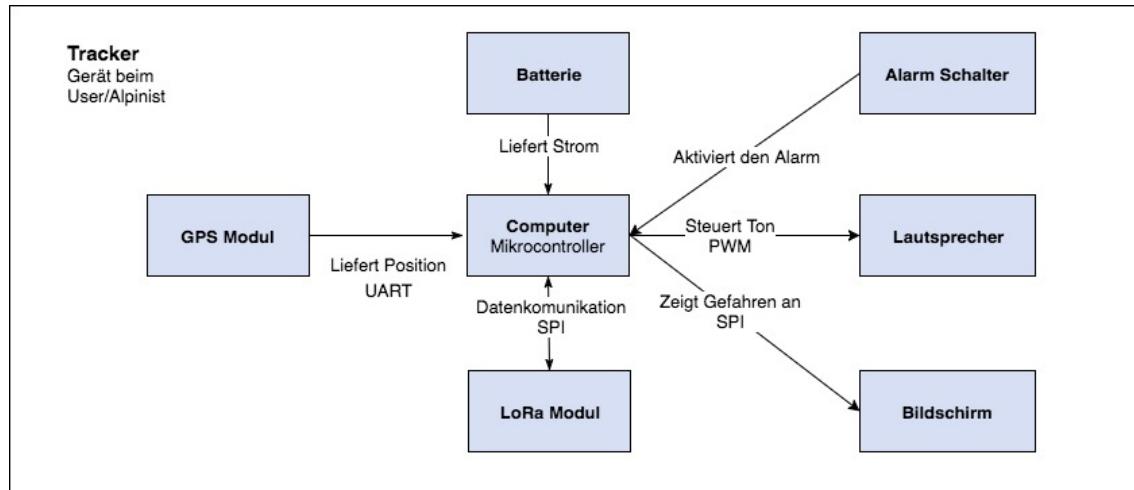


Abbildung 4.4: Tracker Detail

- **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 (LoRaWan Gateway) gesendet werden.
- **Computer:** Verarbeitet eingehenden Signale und löst entsprechende Aktionen aus.

4.2.1 ATAS-Node

ATAS-Node ist die Software welche auf dem Tracker Gerät ausgeführt wird. ATAS-Node wurde, im Vergleich zum Vorprojekt, komplett umgebaut. Genauere Angaben, die ATAS-Node aufgebaut wurde ist im Abschnitt Prototype zu finden.²

4.3 Gateway

Der Gateway verbindet die Tracker mit dem TTN Netzwerk. Er sendet Daten zu den Trackern (Downlink) und empfängt Daten von den Trackern (Uplink).

4.3.1 Diagramm

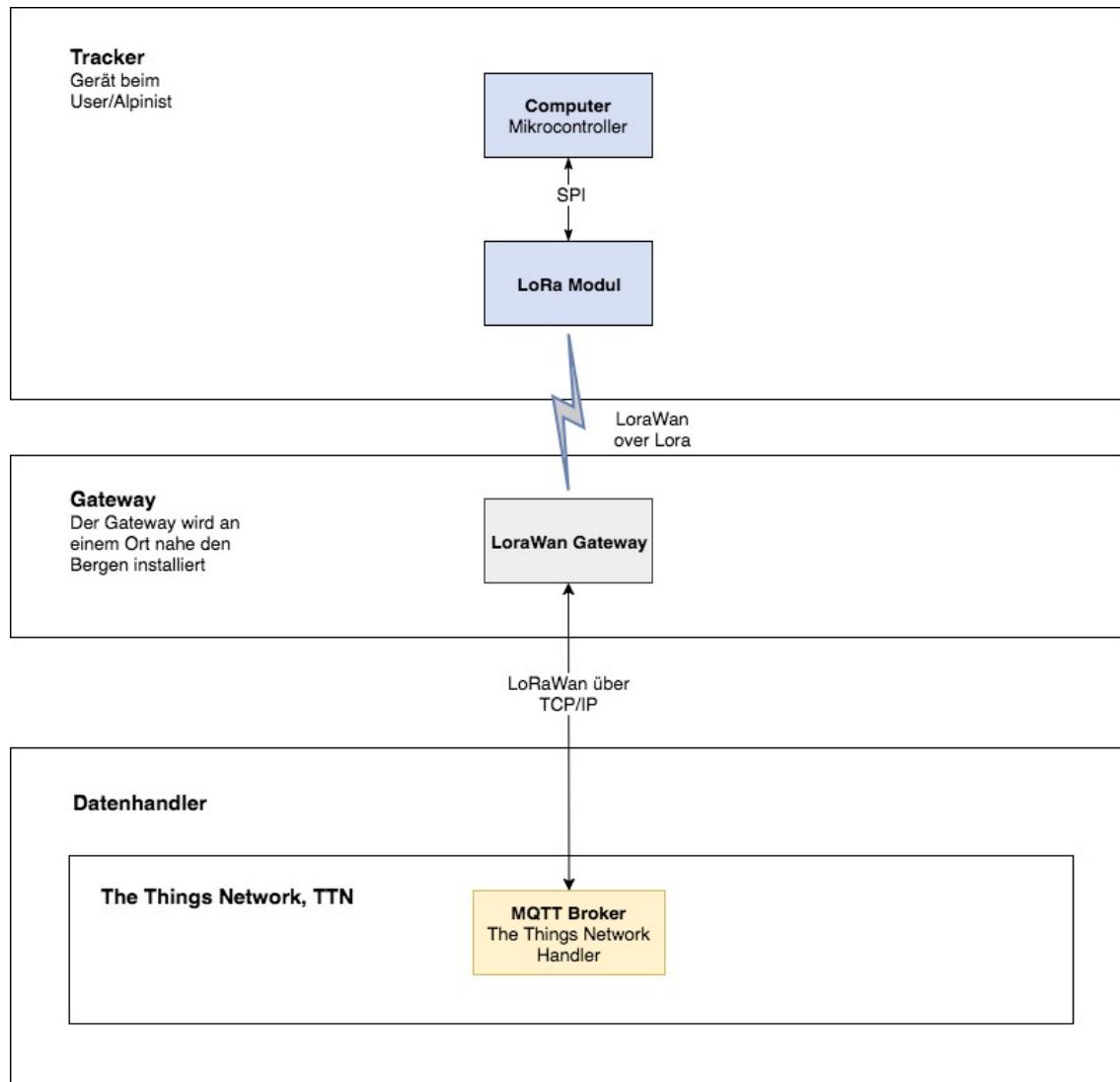


Abbildung 4.5: Gateway Detail

4.4 Datenhandler

Die Aufgabe des Datenhandlers umfasst das Sammeln der Daten und deren Bereitstellung an die ATAS Applikationen ATAS-Web und ATAS-Service. Der Datenhandler besteht aus zwei Komponenten. Die Komponenten wurde nicht selbstständig entwickelt. Für das ATAS Projekt wurden bereits existierende Dienste verwendet. Die Dienste heissen **The Things Network (TTN)** und **CloudMQTT** verwendet. Was genau für Funktionen die einzelnen Dienste bieten wird in den nachfolgenden Abschnitten erklärt.

4.4.1 The Things Network (TTN)

TTN sendet Daten via Gateways an die Tracker. Über die Gateway empfängt das TTN Daten von den Trackern. Das TTN bietet einen Schnittstelle dem MQTT Broker um auf die Daten zuzugreifen.

Diagramm

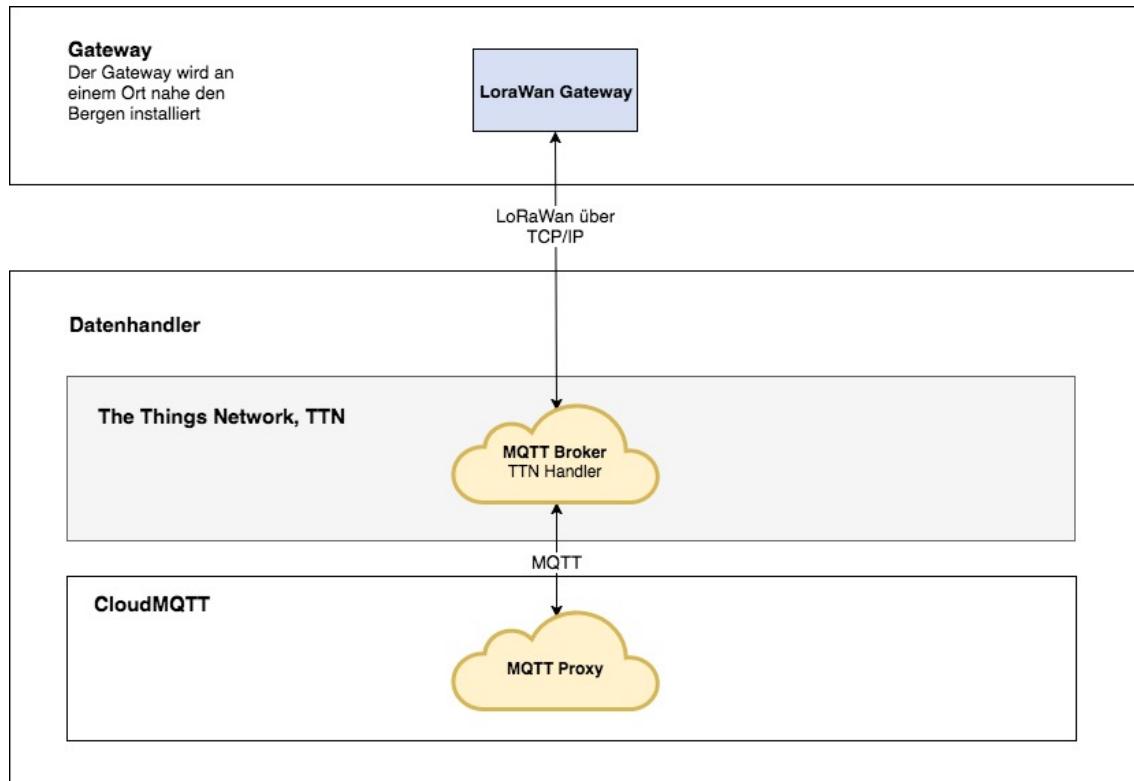


Abbildung 4.6: TTN Detail

4.4.2 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

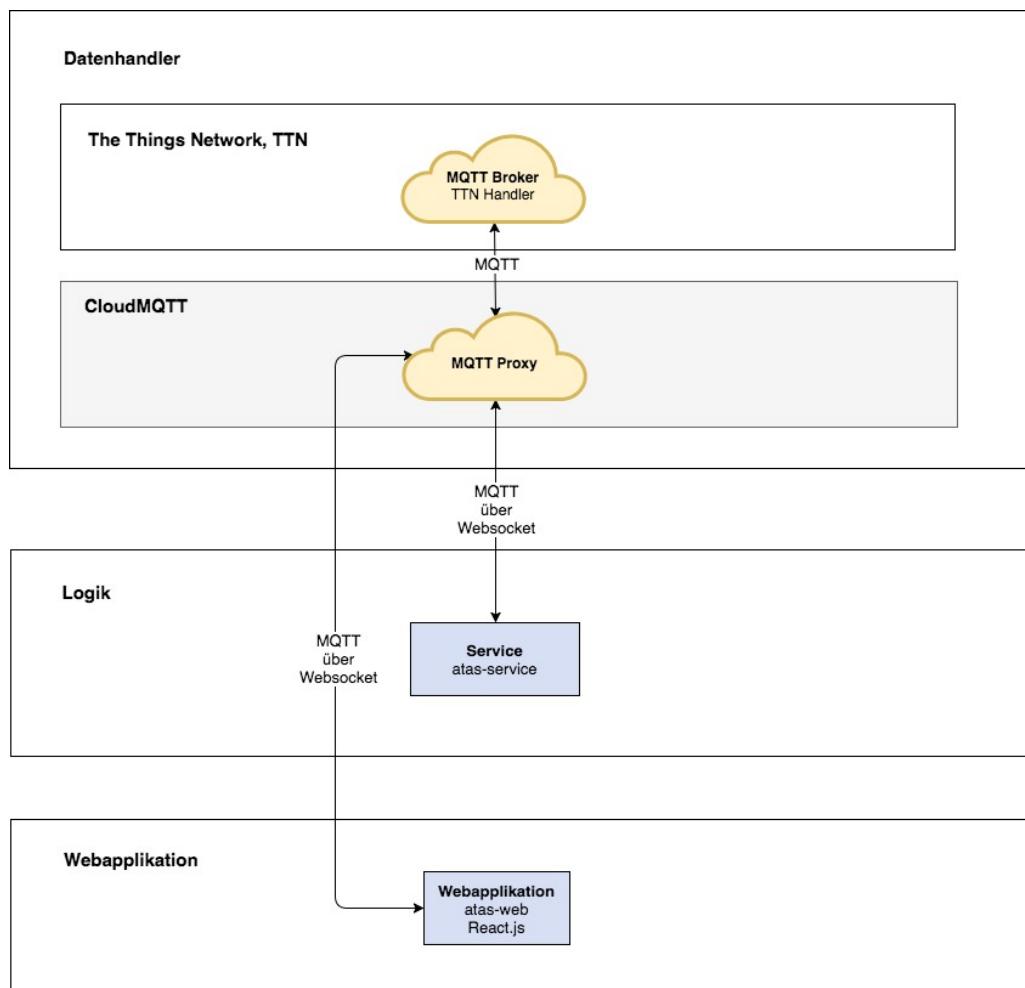


Abbildung 4.7: TTN Detail

4.5 Logik

4.5.1 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.

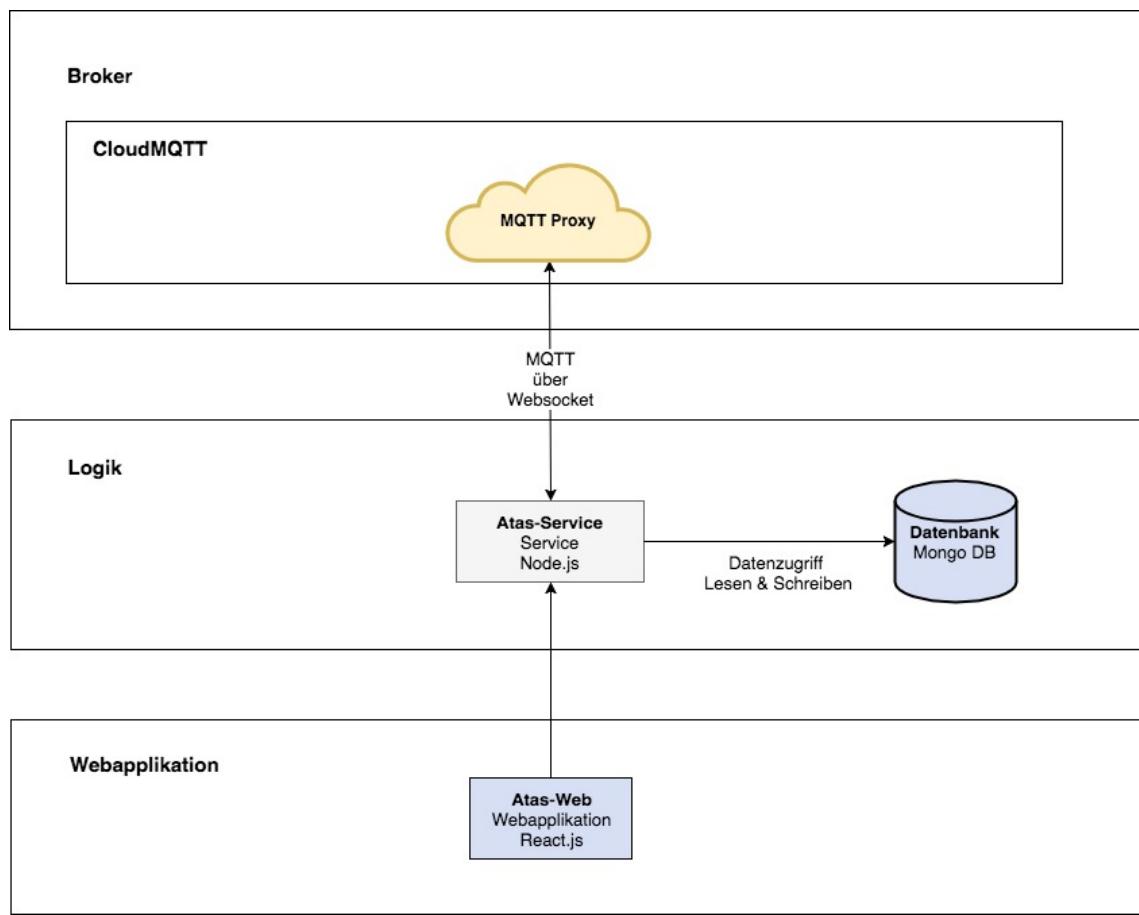


Abbildung 4.8: ATAS-Service Diagram

4.6 Webapplikation

4.6.1 ATAS-Web

ATAS-Web ist eine Webapplikation. Die Applikation wurde mit der Programmiersprache Javascript realisiert. Um die Entwicklung zu beschleunigen wurde auf die Javascript React.js zurückgegriffen. ATAS-Web bildet im wesentlichen das Management Interface für die Rettungsdienste. Über die Applikation kann die aktuelle Position der Tracker auf einer Karte dargestellt werden. Für die Kartendarstellung wurde Google Maps integriert.

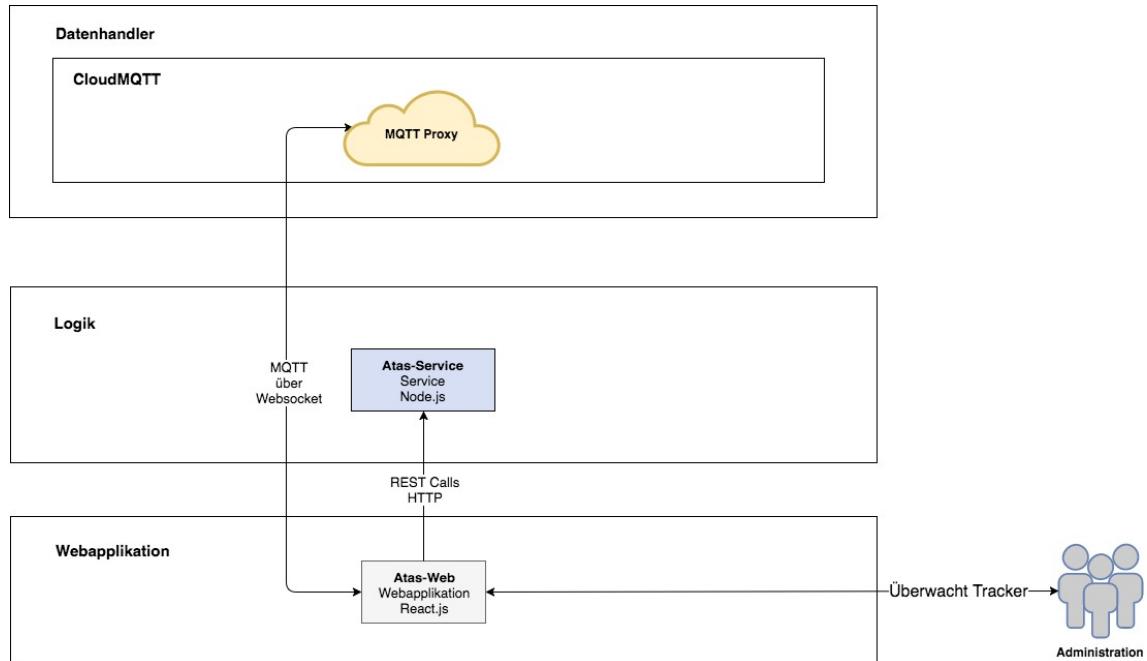


Abbildung 4.9: ATAS-Web Diagramm

4.6.2 Übersicht

Die nachfolgenden Bilder sollen aufzeigen wie ATAS-Web aussieht und wie dieses zu verwenden ist.

Startseite

Die Startseite der Applikation.

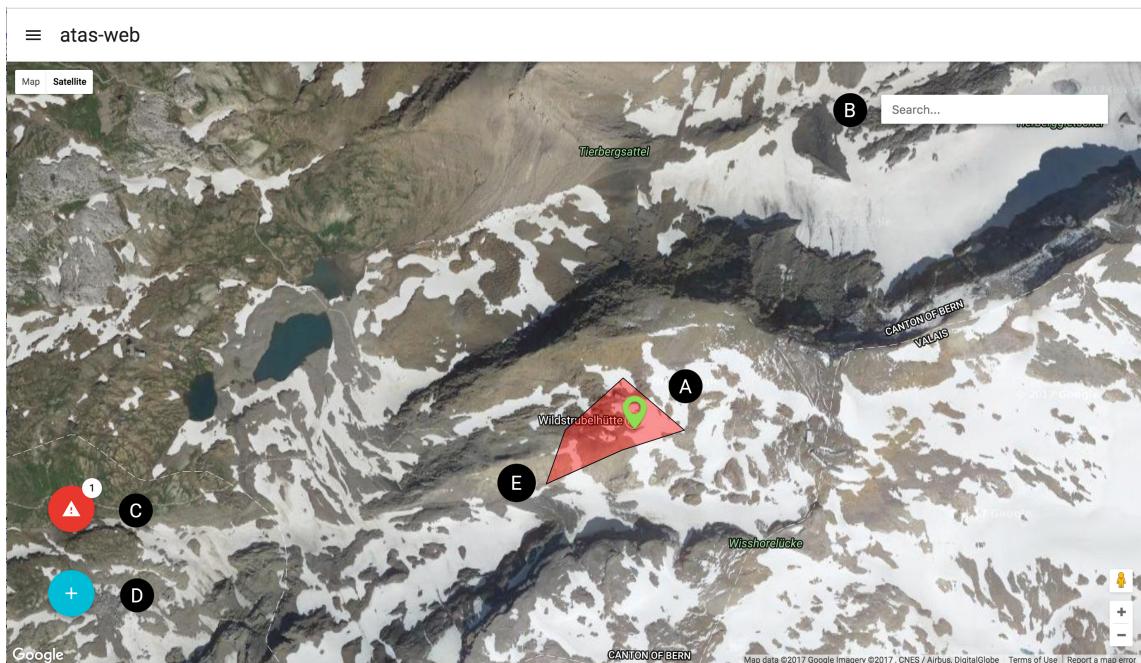


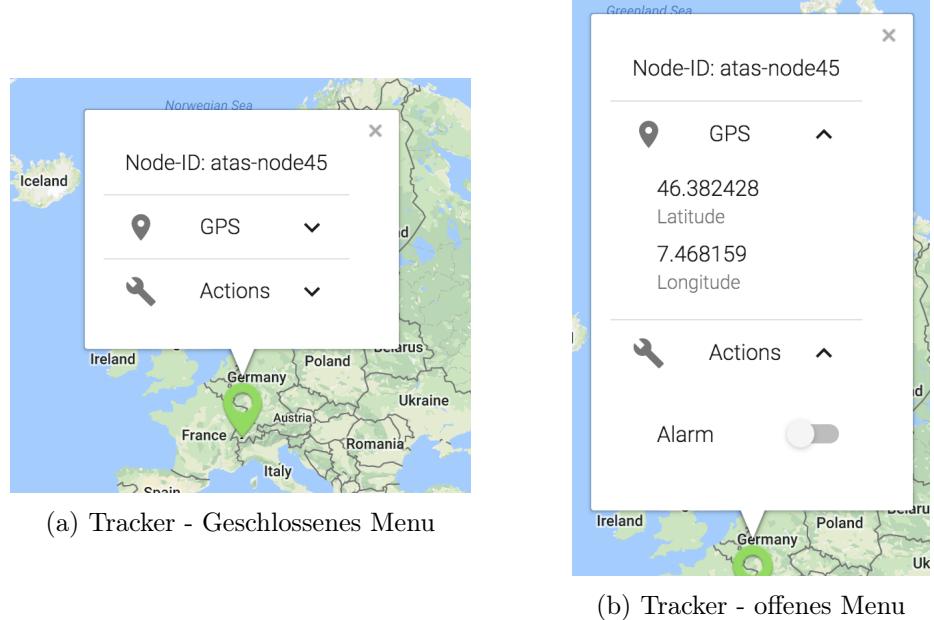
Abbildung 4.10: Startseite

Die Ansicht besteht aus verschiedenen Komponenten

- A: Auf der Karte wird die Position der Tracker dargestellt.
- B: Suche nach Orten
- C: Anzeige der Tracker in Notsituationen
- D: Manuelles hinzufügen von Gefahrenzonen
- E: Gefahrenzone

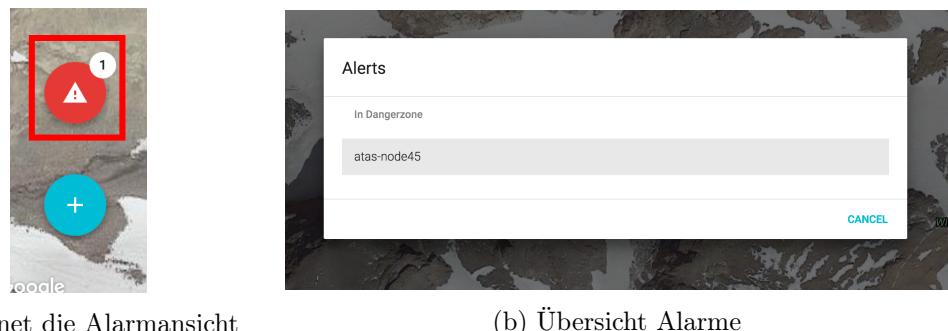
Tracker

Wird ein Tracker ausgewählt öffnet sich ein weiteres Menü. Hier ersichtlich sind nebst der Tracker ID auch die genauen GPS Koordinaten. Über einen Alarmknopf kann ein Alarm auf dem Tracker ausgelöst werden.



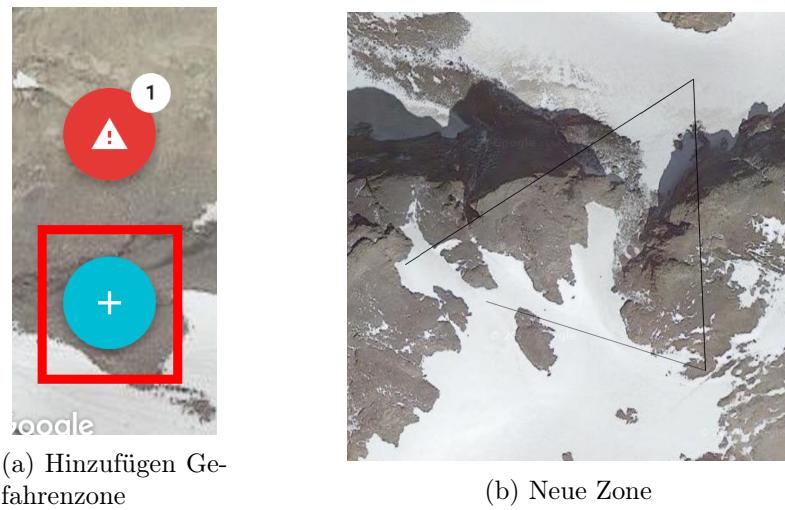
Alarmierung

Über den roten Knopf kann eine neue Ansicht geöffnet werden. Die Ansicht gibt uns eine Übersicht über die Tracker, die sich in einer Gefahrenzone aufhalten oder den manuellen Alarm ausgelöst haben.



Gefahrenzonen

Über den grünen Knopf können manuell Gefahrenzonen angelegt werden. Mittels 'Malwerkzeug' kann eine Polygon gezeichnet werden. Die entstandene Fläche repräsentiert eine Gefahrenzone.



(a) Hinzufügen Ge-
fahrenzone

(b) Neue Zone

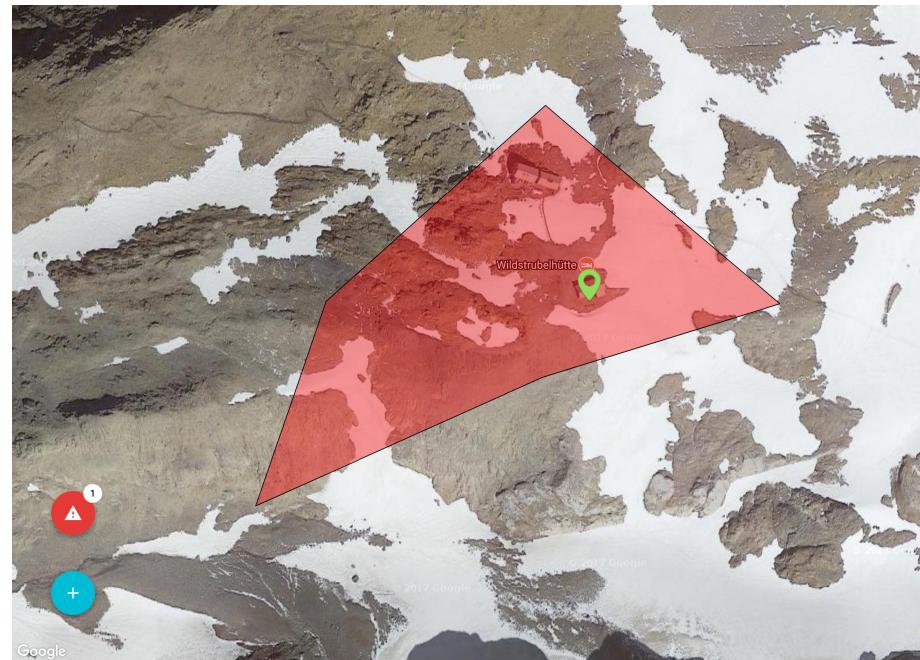


Abbildung 4.14: Gefahrenzone

4.7 Kommunikation

In diesem Abschnitt wird die Kommunikation zwischen den Komponenten im System aufgezeigt.

4.7.1 MQTT

Für das ATAS Projekt wurde die nachfolgende MQTT Struktur verwendet.

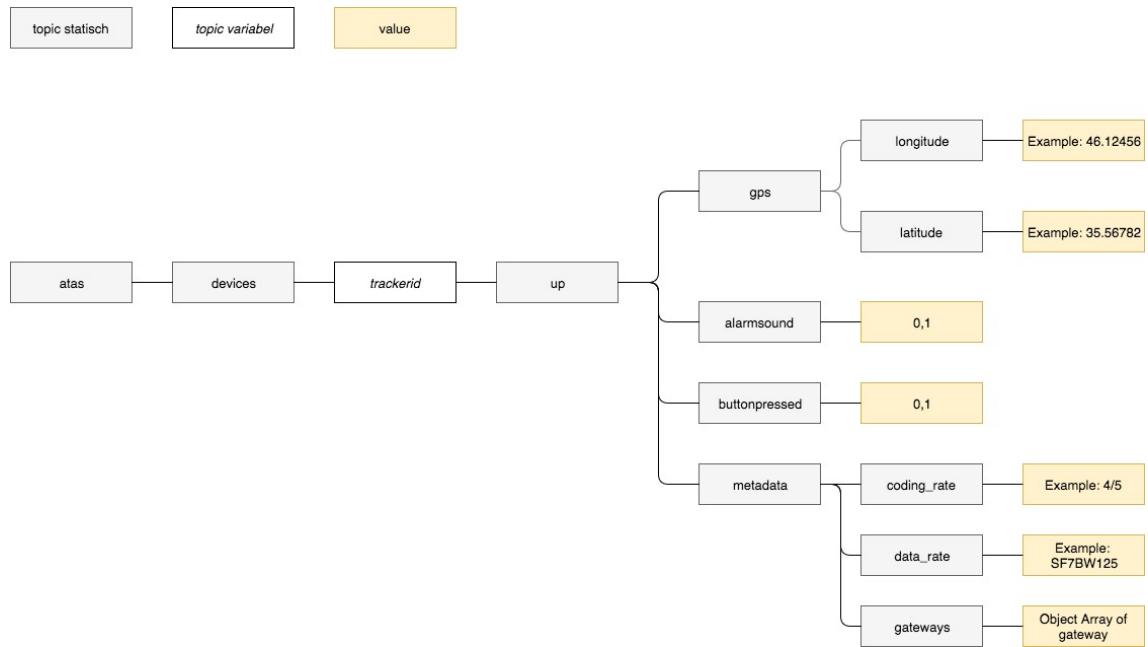


Abbildung 4.15: MQTT Struktur

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 den Bau eines neuen Prototypen

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.

3. Die Verbauten Komponenten sind nicht für eine Industrialisierte Lösung geeignet. 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:

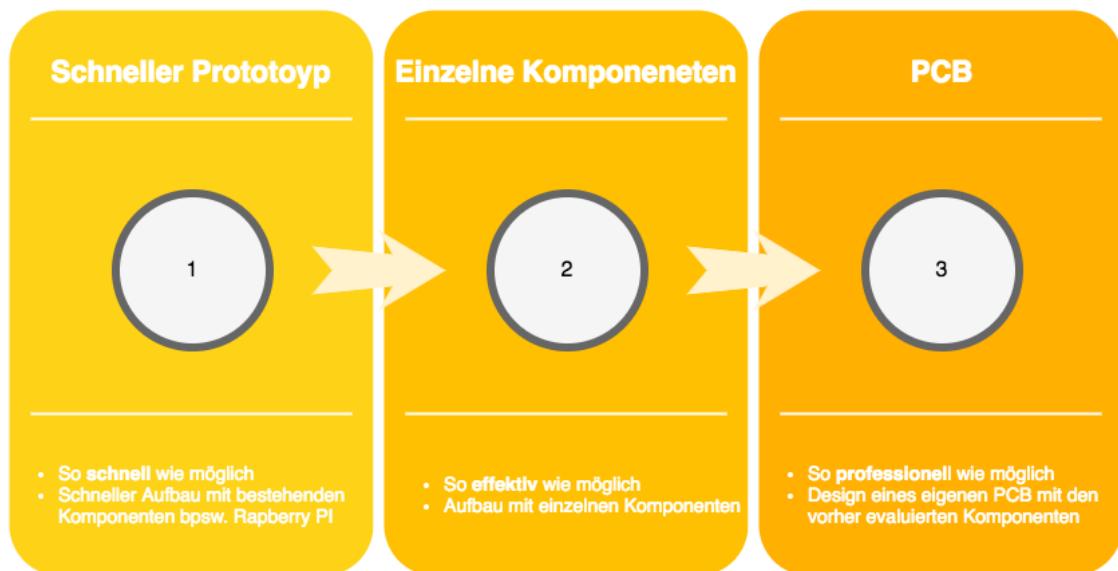


Abbildung 5.1: Prototyping Ablauf

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 funktionsfähigen Prototypen aufzubauen. Während dieser Arbeit sollen nun neue Komponenten evaluiert und ein zweiter Prototyp erstellt werden. Der Ablauf dieser Phasen sieht wie folgt aus:

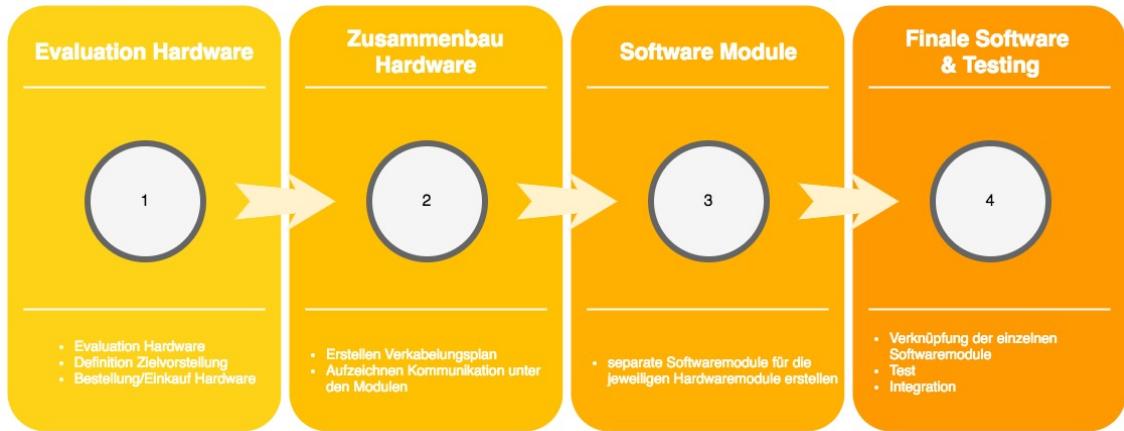


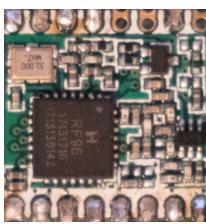
Abbildung 5.2: Ablauf für den Aufbau des zweiten Prototyps

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



Es gab kein spezielles Auswahlverfahren für die Komponenten. Die Auswahl wurde durch Internetrecherchen und Empfehlungen von Kollegen beeinflusst. Bei einem erneuten Projekt in dieser Richtung müsste ich vorher die Kriterien für die Komponenten festlegen.

5.4.1 Lora-Antenne

Es wurde angenommen, dass beim Kauf des Lora-Moduls eine Antenne mitgeliefert wird. Leider war dies nicht der Fall. Dieser Zwischenfall stellte eine gute Gelegenheit dar, das Wissen über die Telekommunikationswelt zu vertiefen. Es wurde entschied keine erneute Bestellung auszulösen und stattdessen die Antenne einfach selbstständig zu bauen. Nach einer kurzen Internetrecherche stellte sich heraus, dass mit einfachen Mitteln eine zuverlässige Antenne für 868Mhz gebaut werden kann [18]. 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 [19].

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 den Prototypen zu lange. Es wurde entschieden, 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:

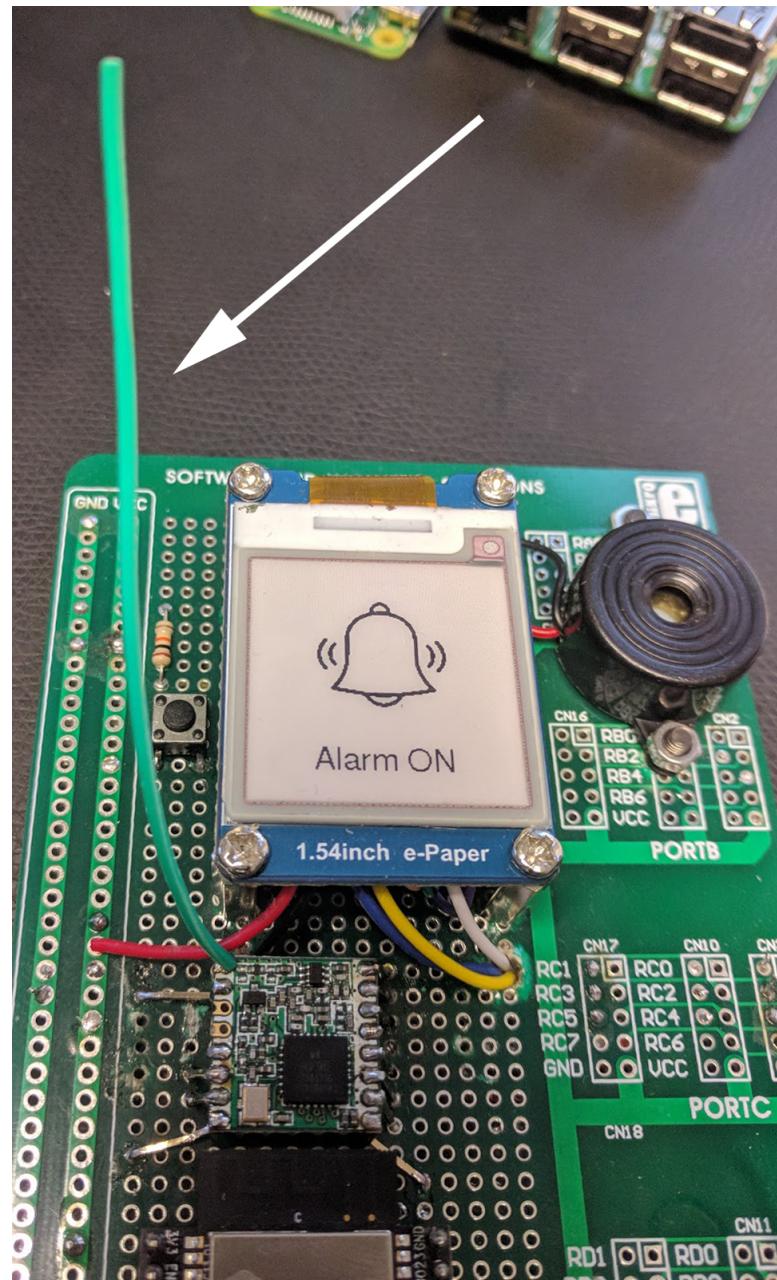


Abbildung 5.3: Prototyp 2 Antenne]

Die Kommunikation funktioniert einwandfrei und ist mit der des ersten Prototypen vergleichbar. Der Empfang würde mit einer professionellen Antenne noch besser werden. 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 immer unübersichtlicher. Um mir die Arbeit zu erleichtern, wurde 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 [24]. 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.

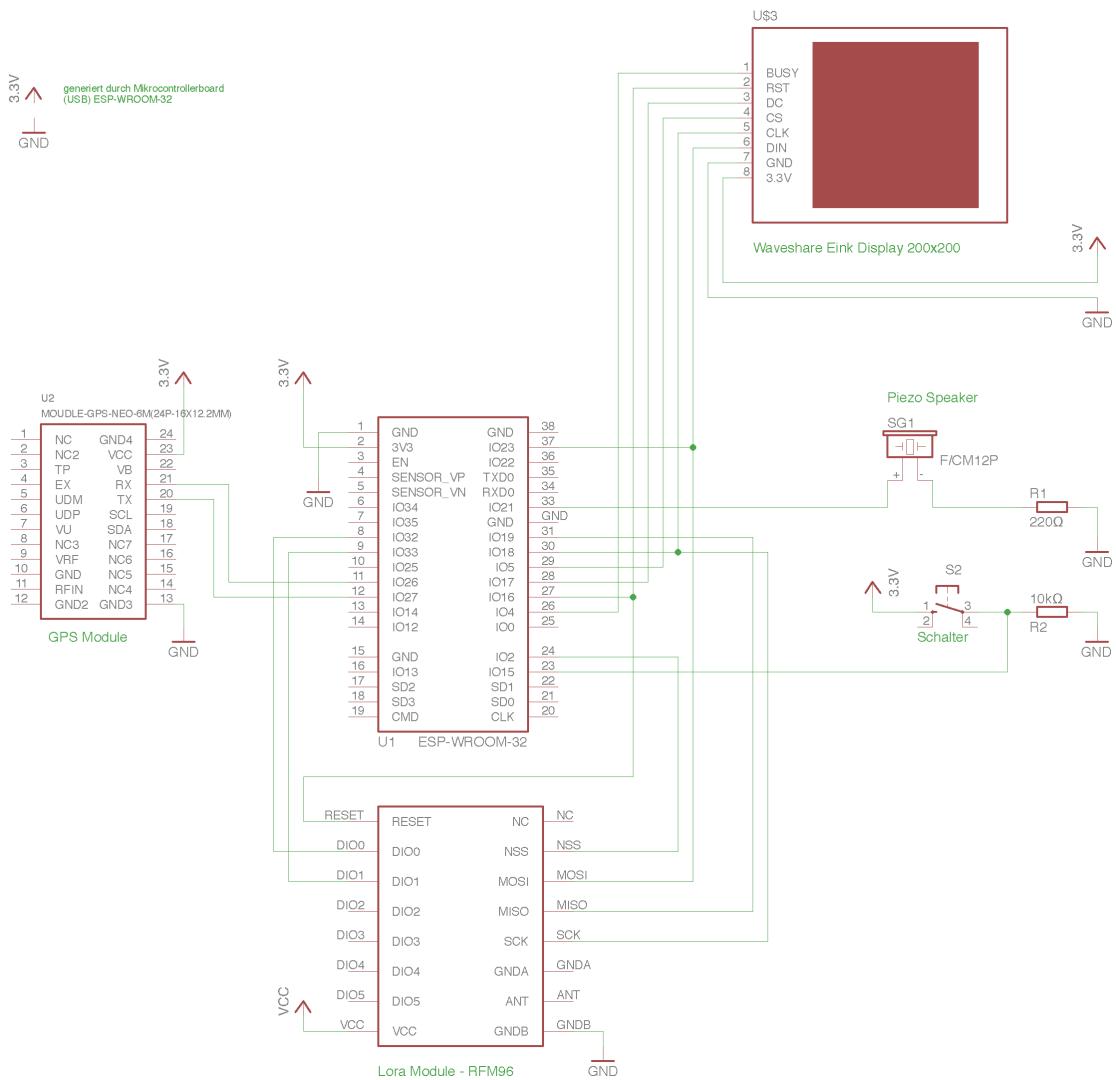


Abbildung 5.4: Prototyp 2 Schema

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	Displayl	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.

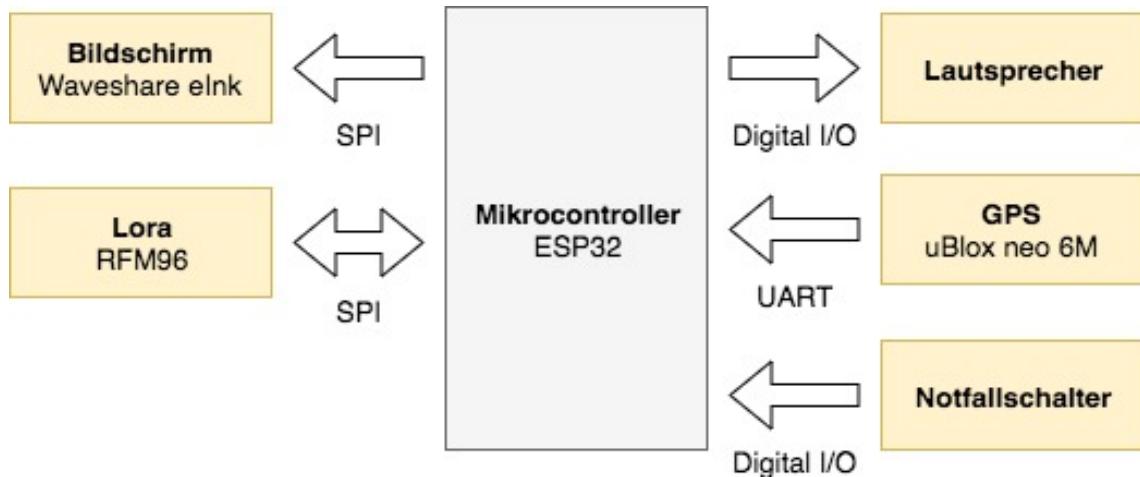


Abbildung 5.5: Kommunikationsart Prototyp 2

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.

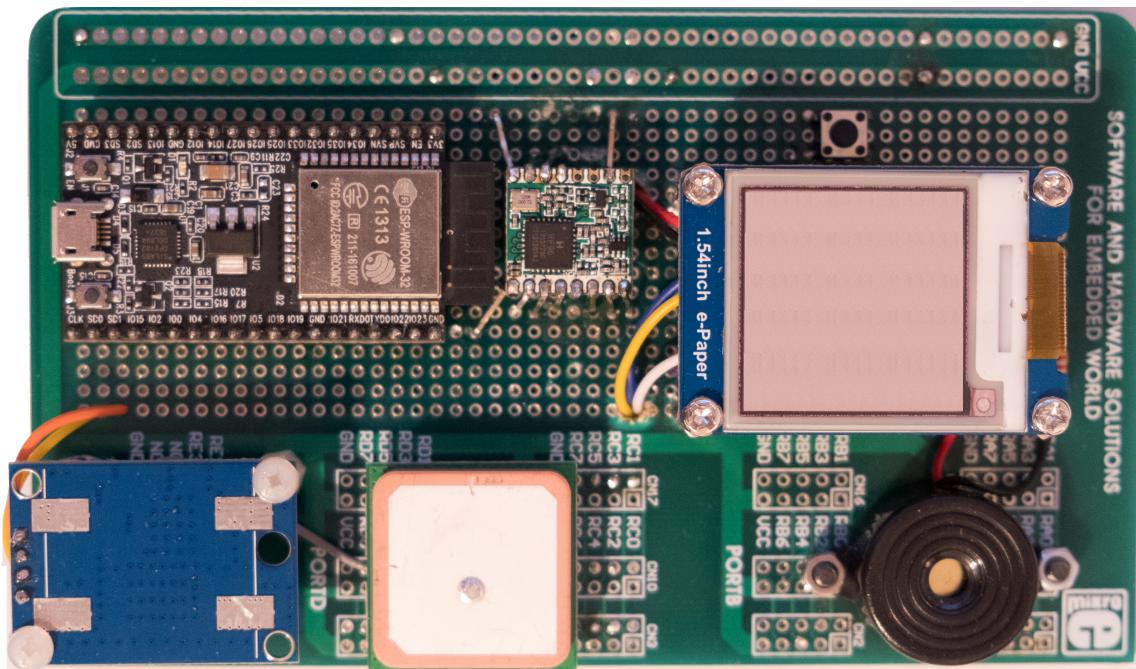


Abbildung 5.6: Prototyp 2

5.7 Software

Der bestehende Sourcecode des Prototyp 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 nachfolgenden Abschnitte beschreiben den Aufbau der Software ATAS-Node welche für den neuen Prototypen angepasst wurde. Die Logik der Software konnte grösstenteils übernommen werden.

5.7.1 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[25]. 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. In Absprache mit dem Betreuer wurde festgelegt, dass die ESP-IDF Umgebung zu verwenden sei. Die Installation der Entwicklungsumgebung ist nicht sonderlich schwer, bedingt aber einige manuelle Schritte. Die Umgebung besteht aus einer Reihe von Tools (Toolchain) zum Kompilieren, Fehlersuche und dem Flashen des Mikrocontrollers.

An dieser Stelle wird auf eine detaillierte Auflistung aller Installationsschritte verzichtet. Die Installation wurde gemäss der Anleitung auf der Webseite der Firma Espressif durchgeführt [26].

Als Hostsystem wurde ein macOS System verwendet. Damit das System den ESP via USB korrekt erkennen konnte, musste ein zusätzlicher Treiber installiert werden [27].

5.7.2 Sourcecode

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

5.7.3 Bibliotheken

Infolge der Umstellung auf die ESP32 Umgebung mussten einige bestehenden Bibliotheken ersetzt werden. Die verwendeten Software Bibliotheken werden im Abschnitt 'Anhang' aufgeführt.

LMIC

Die LMIC Bibliothek hat den LoRaWan Stack implementiert und wird für die korrekte Kommunikation via LoRa Modul benötigt. Zum Zeitpunkt dieser Arbeit gibt es keine für ESP32 portierte LMIC Bibliothek. Für die LoRaWan Kommunikation wurde daher auf eine bestehende Arduino Library (Arduino-LMIC) zurückgegriffen [8]. Der Funktionsumfang bleibt der selbe. Den Kommunikationsablauf selbstständig zu programmieren hätte den Umfang dieser Arbeit gesprengt. Einige Entwickler berichten von Timing Problemen während der Kommunikation. Bis jetzt konnten beim ATAS Projekt keine Probleme mit der eingesetzten Library festgestellt werden. Die Arduino Library fügt sich problemlos in die ESP-IDF Umgebung ein.

5.7.4 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.

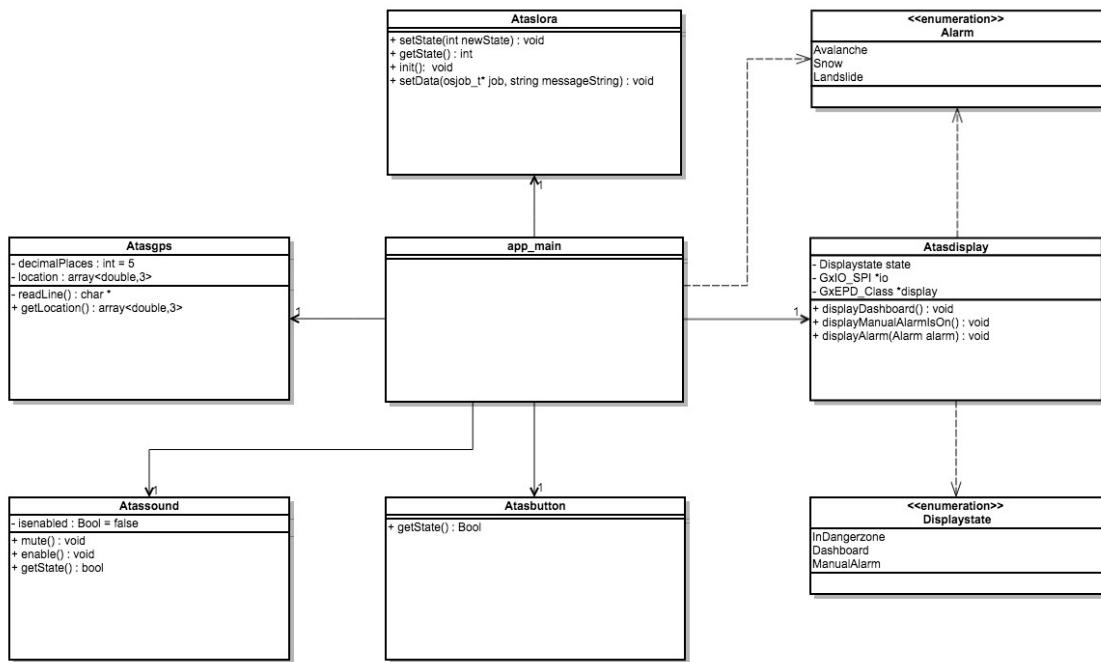


Abbildung 5.7: Klassendiagramm

5.7.5 Display

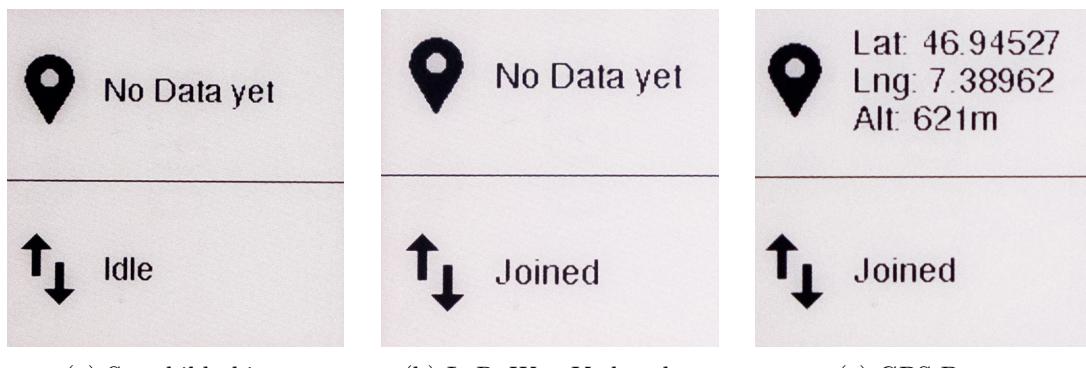
Um die Alarmierung des Alpinisten zu verbessern wurde beim Tracker ein Display eingebaut. Um die Batterie des Trackers nicht weiter zu belasten wurde ein eInk Display ausgewählt. Das Display benötigt nur Energie wenn das angezeigte Bild ändert, siehe Abschnitt 'Testing'.

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, durch einen Druck auf den Notfallknopf, selbst ausgelöst. Der Bildschirm dient der Bestätigung für das erfolgreiche aktivieren des Alarms.



Abbildung 5.10: Notfallalarm

Herausforderung Timing

Die Schaltung des Prototypen wurde so aufgebaut, dass möglichst wenig Kabel gezogen werden mussten. Infolge dessen wurden der Bildschirm und das LoRa Modul über einen einzigen SPI Kontroller angesteuert. Relativ schnell machte sich ein Timing Problem bemerkbar. Die Software versuchte das Display zu aktualisieren und im gleichen Moment Daten per LoRa versenden. Die Kommunikation mit dem Bildschirm wurde gestört, der Bildaufbau wurde unterbrochen

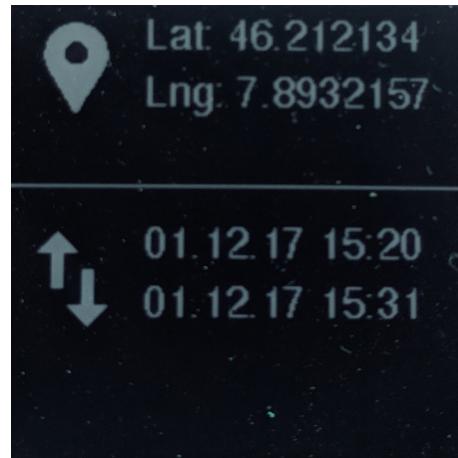


Abbildung 5.11: Fehlerhafter Bildaufbau Display

Um die korrekte Datenübertragung sicherzustellen, wurde in der Software ein Mutex implementiert. Die Ansteuerung des Bildschirm und des LoRa Kommunikation wurde in zwei unabhängige Tasks aufgeteilt. Ist ein Task beschäftigt Daten zu senden, wird die Mutex Ressource gesperrt. Der andere Tasks muss warten, bis er wieder Daten senden kann. Mit dieser Lösung kann sichergestellt werden, dass es zukünftig keine Kommunikationskonflikte mehr geben wird.

```
if(xSemaphoreTake( xSPISemaphore, portMAX_DELAY ) == pdTRUE){
    // redraw
    atasdisplay->updateDisplay();
    // make sure picture is all drawn
    delay(3000);
    xSemaphoreGive(xSPISemaphore);
}
```

(a) Mutex - Bildschirm

```
// get semaphore
if(xSemaphoreTake( xSPISemaphore, portMAX_DELAY ) == pdTRUE){
    // run, until the next tx succeeds
    while(ataslora->getSendState() == send){
        os_runloop_once();
        delay(1);
    }
    xSemaphoreGive(xSPISemaphore);
}
```

(b) Mutex - Lora

5.8 Finalisierung & Testing

Abschließend werden 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 resp. Systemzustände sollen geprüft werden.

- Ruhezustand
- Daten senden via LoRa
- Bildschirm aktualisieren
- Lautsprecher ein

Ausgangslage: Das Gerät wird an ein Speisegerät angeschlossen. Die Betriebsspannung beträgt bei beiden Geräten 5V. Die beiden Prototypen verfügen über einen eingebauten Spannungswandler welche die Spannung auf 3.3V herunter regelt. Die dabei entstehenden Verlustleistung wird bei den nachfolgenden Tests ignoriert. Während des Tests sind alle Module (LoRa, GPS...) angeschlossen, werden also mit Strom versorgt. Der Prototyp 1 verfügt über kein Display, deshalb können keine Daten für diesen Systemzustand erhoben werden. Die entsprechenden Szenarien werden beim Prototypen eingesetzt und die benötigte Leistung werden vom Speisegerät abgelesen.

Messresultate:

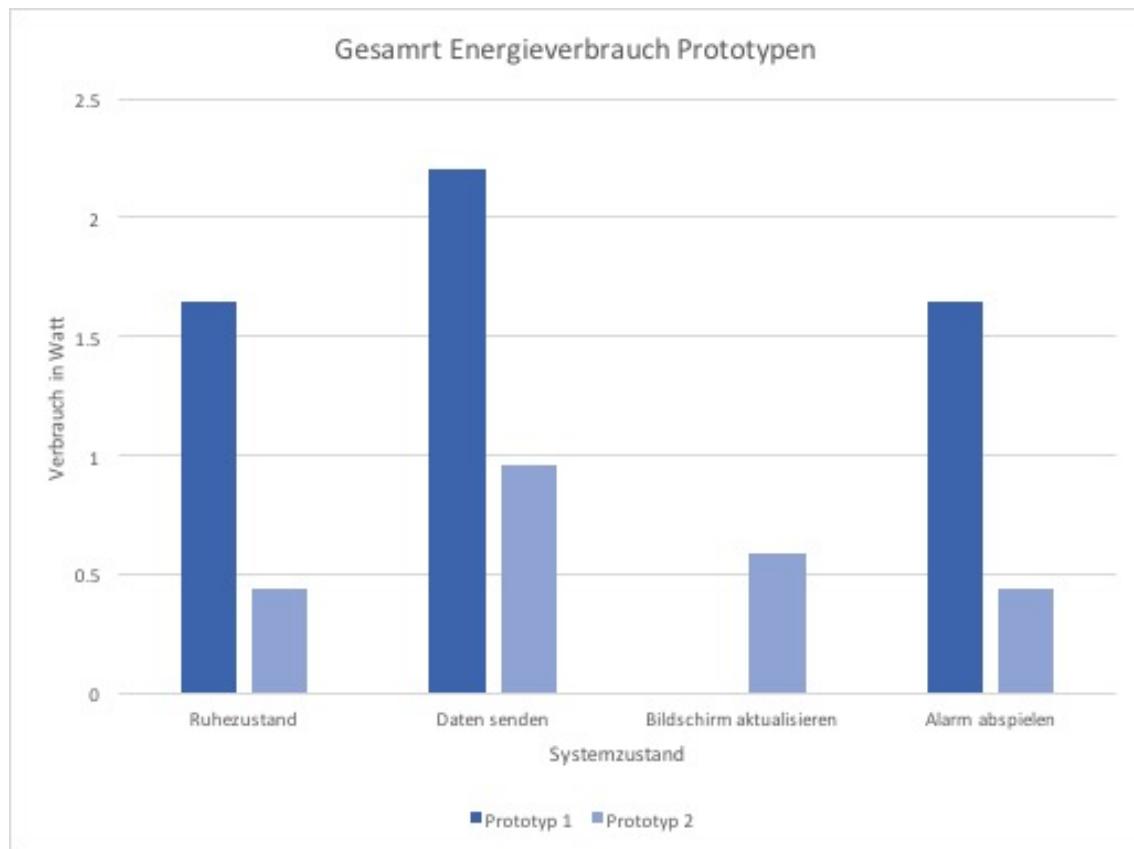


Abbildung 5.13: Gesamt Energieverbrauch pro Prototyp

Diese Grafik zeigt den gesamten Energieverbrauch des Systems während der durchgeführten Aktion. Wie in der Grafik ersichtlich, benötigt der Prototyp 1 im Ruhezustand (0.4W) ca. 4 mal soviel Energie wie der neue Prototypen (1.6W). Durch den Wechsel der Hardware konnte der Energieverbrauch drastisch gesenkt werden.

Das senden der Daten lässt den Energieverbrauch stark ansteigen. Beim Prototypen 2 wächst der Verbrauch gar um 100%. Das senden via LoRa macht also einen großen Teil des Energieverbrauchs aus.

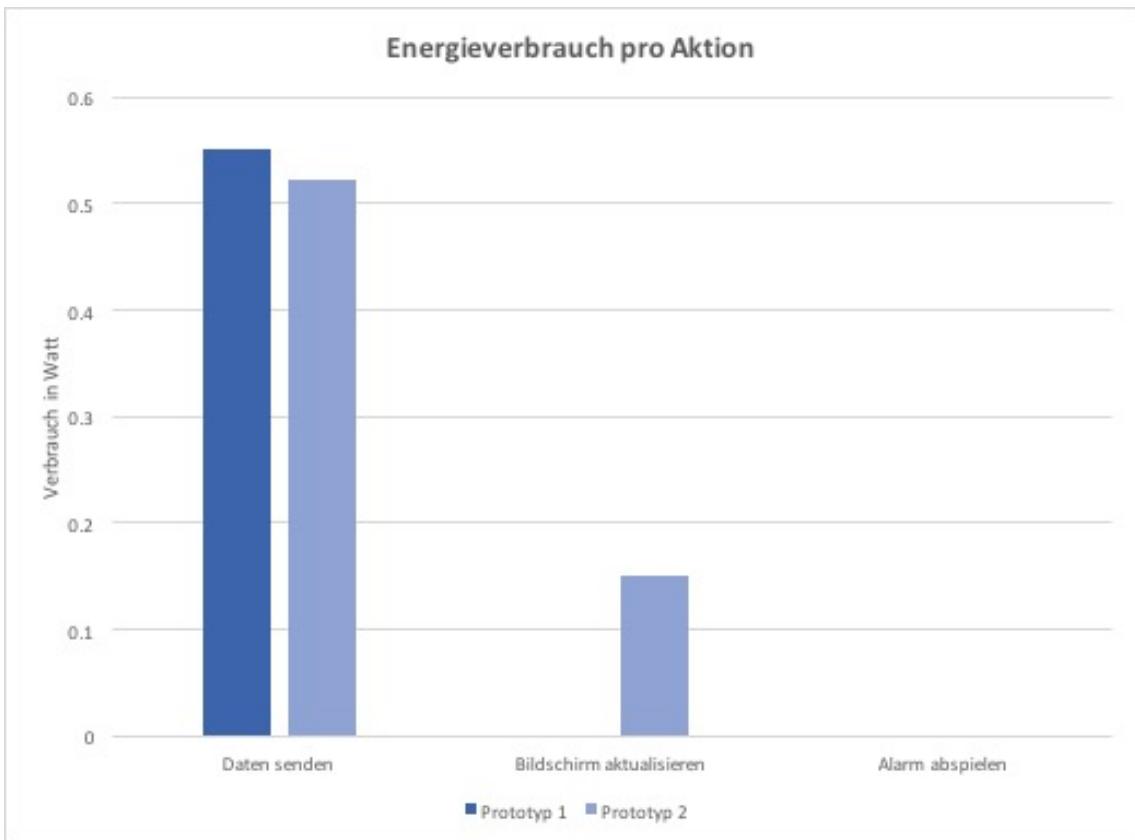


Abbildung 5.14: Veränderung des Energieverbrauchs zum Ruhezustand

Der Gesamtverbrauch des Systems während einer Aktion wurde mit dem Verbrauch des Ruhezustandes subtrahiert. Diese Grafik zeigt nun wie die einzelnen Aktionen den Energieverbrauch beeinflussen. Die nachfolgenden Erkenntnisse wurden daraus gewonnen

- Da auf beiden Prototypen der gleiche LoRa Chip verbaut ist, ist es keine Überraschung, dass die benötigte Energie in etwa identisch ist.
- Das Aktualisieren des Displays benötigt, im Vergleich zum Senden via LoRa, sehr wenig Energie.
- Das Aktivieren des Piezo Lautsprechers hat keinen messbaren Einfluss auf den Energieverbrauch.

5.8.2 Schlusswort

Die Resultate mit dem neuen Prototypen sind zufriedenstellend. Das Display erweitert den Prototypen um ein wichtiges Mittel der Alarmierung. Das System konnte in Sachen Energiebedarf optimiert werden. Der Prototyp 2 funktioniert einwandfrei. Dem nächsten Schritt, dem finalen Prototypen steht somit nichts mehr im Weg.

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 dem Tracker (Atas-Node) und dem 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.

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

Diese Ansicht gibt uns einen guten Überblick. Leider müssen die Daten alle manuell herauskopiert werden um diese analysieren zu können. Es gibt momentan keine Möglichkeit die gesamten Daten von TTN herunterzuladen. Diese Tatsache macht die Auswertung der Daten bspw. mit Excel sehr umständlich.

- 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

Wir fassen zusammen

- Es muss eine Testumgebung aufgebaut und die Verbindung getestet werden. Um von den bestehenden TTN Gateways unabhängig zu sein, wird ein **eigener Gateway** aufgebaut.
- Da das TTN Dashboard uns keine vernünftige Lösung bietet die Daten auszuwerten, soll eine eigene Lösung entwickelt werden. Die Komponenten Atas-Web und Atas-Service müssen entsprechend erweitert werden.
- Abschließend werden mehrere Tests durchgeführt. Die Verbindung wird aufgebaut und mittels neu entwickeltem Dashboard geprüft.

6.3 Standort

Ausgehend von den vorangegangen Definitionen muss folgendes definiert werden.

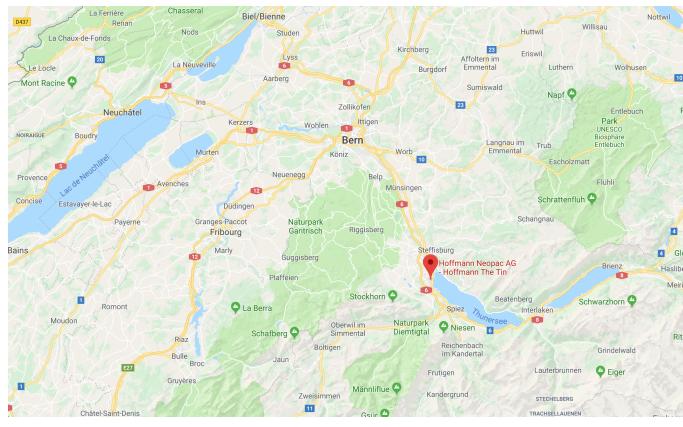
- Es muss ein Ort gefunden werden, wo der Gateway platziert werden kann.

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. 2 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 Thun. The map shows various towns, lakes (e.g., Lake Thun, Lake Brienz), and mountains. A red dot marks the location of Hoffmann Neopac AG in Thun.
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.

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	15Km

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 [15]
- C: IP67-Box

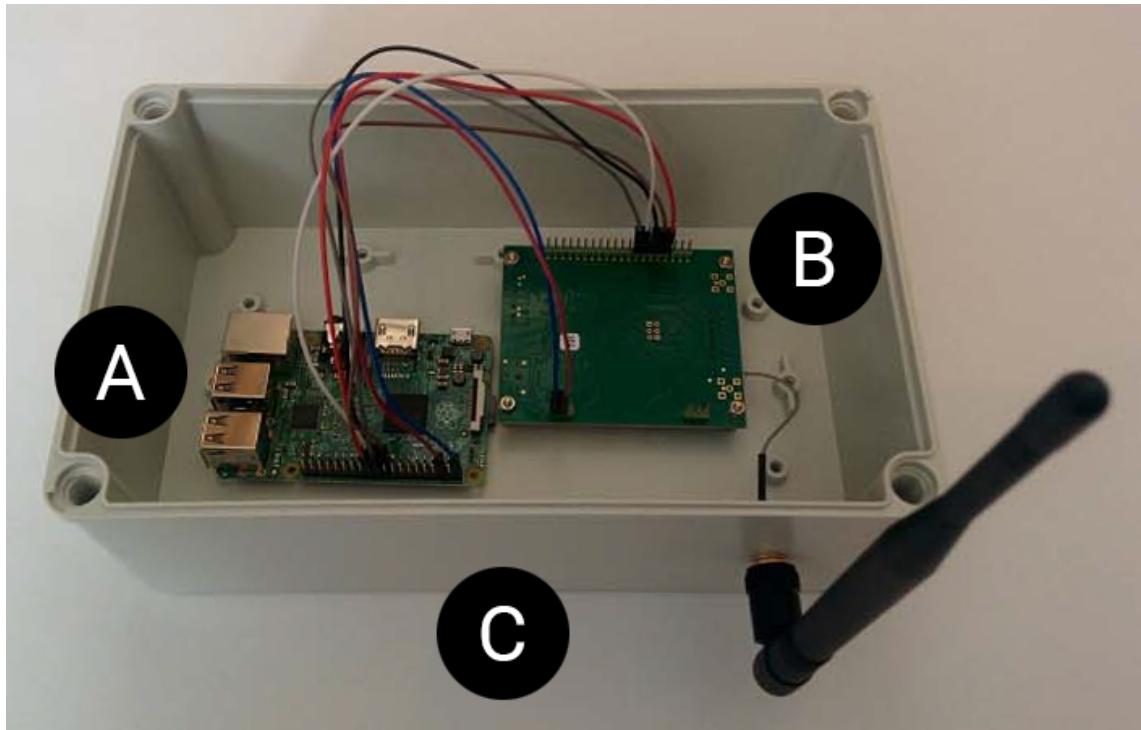


Abbildung 6.3: Gateway

Durch den Einsatz einer IP67 Box sind die elektronischen Komponenten gegen Staub und Wasser geschützt. Leider wurde hier zu wenig Zeit in die Planung investiert. Wie sollte der Pi mit Strom versorgt werden, 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 [14]. 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.

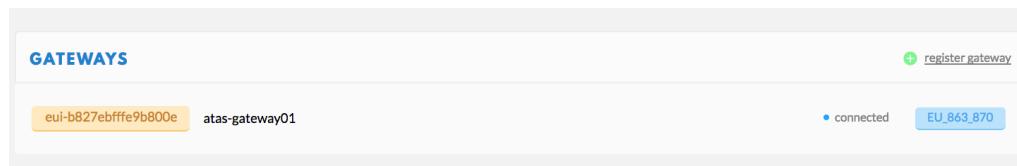


Abbildung 6.5: TTN Dashboard Gateway Registriert

6.4.2 Dashboard

Das eingeschränkte TTN Dashboard genügt für das Testing nicht. In diesem Abschnitt wird erklärt, wie die ATAS Software erweitert wurde, um die Testdaten besser analysieren zu können.

ATAS-Web

Die Webkomponente ATAS-Web wurde um einige Funktionen erweitert. Über den Menüpunkt 'Data' kann das Dashboard aufgerufen werden.



Abbildung 6.6: ATAS-Web Menu

Das Dashboard bietet folgende Funktionen.

- A: Über ein Dropdown kann der entsprechende Tracker ausgewählt werden.
- B: Da eine Uplink Nachricht von mehreren Gateways empfangen werden kann, ist es nötig die Nachrichten nach Gateway zu filtern.
- C: Die aktuell angezeigten Daten können als CSV Datei heruntergeladen werden
- D: das Dashboard zeigt die empfangenen Uplink Nachrichten an

atas-web

Date	Time	Counter	Latitude	Longitude	Alert Triggered	In Dangerzone	RSSI	SNR	Gateway	DataRate	CodingRate	ID
11.01.20...	20:39	6	999	999	false	false	-118	-2	eui-0080...	SF7BW1...	4/5	5a57bd7...
11.01.20...	20:39	5	999	999	false	false	-120	-5.2	eui-0080...	SF7BW1...	4/5	5a57bd5...
11.01.20...	20:38	3	999	999	false	false	-118	-0.2	eui-0080...	SF7BW1...	4/5	5a57bd2...
11.01.20...	20:37	2	999	999	false	false	-119	-5.5	eui-0080...	SF7BW1...	4/5	5a57bd1...
11.01.20...	20:37	1	999	999	false	false	-117	-7.8	eui-0080...	SF7BW1...	4/5	5a57bce...
11.01.20...	20:35	9	47.14265	7.24412	false	false	-119	-4.5	eui-0080...	SF7BW1...	4/5	5a57bc8...
11.01.20...	20:34	7	47.14288	7.24409	false	false	-117	-8.5	eui-0080...	SF7BW1...	4/5	5a57bc5...
11.01.20...	20:32	3	47.14274	7.24423	false	false	-118	-9	eui-0080...	SF7BW1...	4/5	5a57bbe...
11.01.20...	20:31	1	47.14274	7.24423	false	false	-119	-6.5	eui-0080...	SF7BW1...	4/5	5a57bb9...
11.01.20...	20:27	4	47.14277	7.2441	false	false	-119	-7	eui-0080...	SF7BW1...	4/5	5a57ba9...
11.01.20...	20:25	24	47.14344	7.24297	false	false	-120	-5	eui-0080...	SF7BW1...	4/5	5a57ba1...
11.01.20...	20:25	23	47.14344	7.24297	false	false	-119	-6.2	eui-0080...	SF7BW1...	4/5	5a57ba0...
11.01.20...	20:24	22	47.14344	7.24297	false	false	-117	-6.2	eui-0080...	SF7BW1...	4/5	5a57ba0...

Abbildung 6.7: ATAS-Web Dashboard

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 4/8

Resultate

TODO, PRIO 1

6.5.2 Test 2: Spreadingfactor

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

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

Test 2.1

Ausgangslage: 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. Der kleinste wie auch der größte Messwert werden entfernt. Damit soll verhindert werden, dass Ausreißer die Statistik stören. Damit bleiben uns 8 Messpunkte.

Annahme: Gemäß Literatur müsste klar erkennbar werden, dass bei einem höheren SF Wert, das Signal mit weniger Störung übertragen werden kann, dH. Das Signal stärker beim Gateway eintrifft.

Die Messresultate:

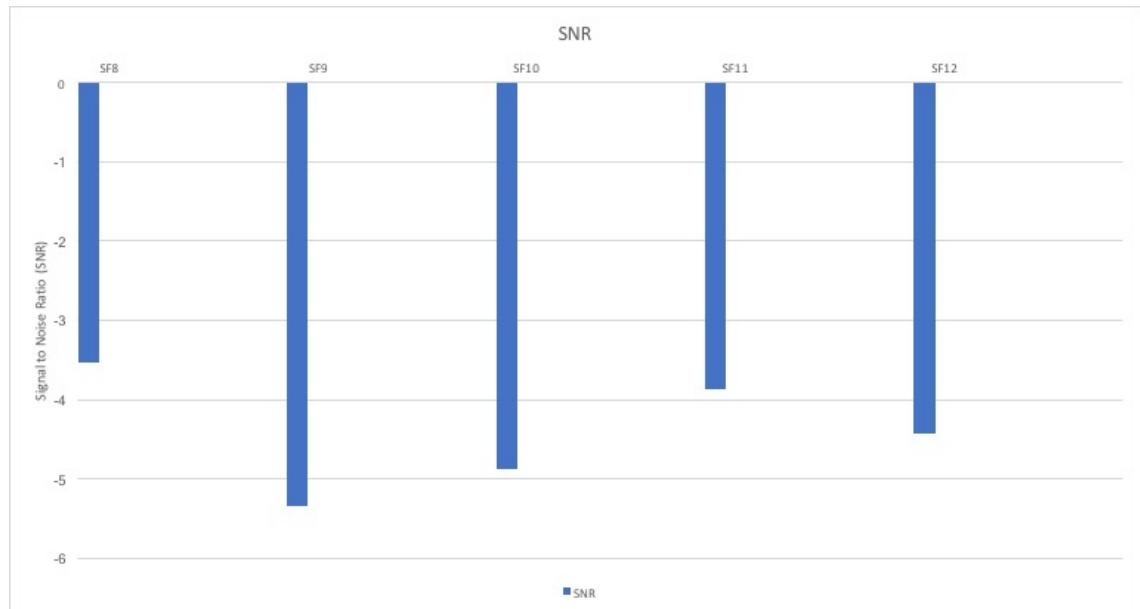


Abbildung 6.8: Messwerte SNR pro Spreadingfactor

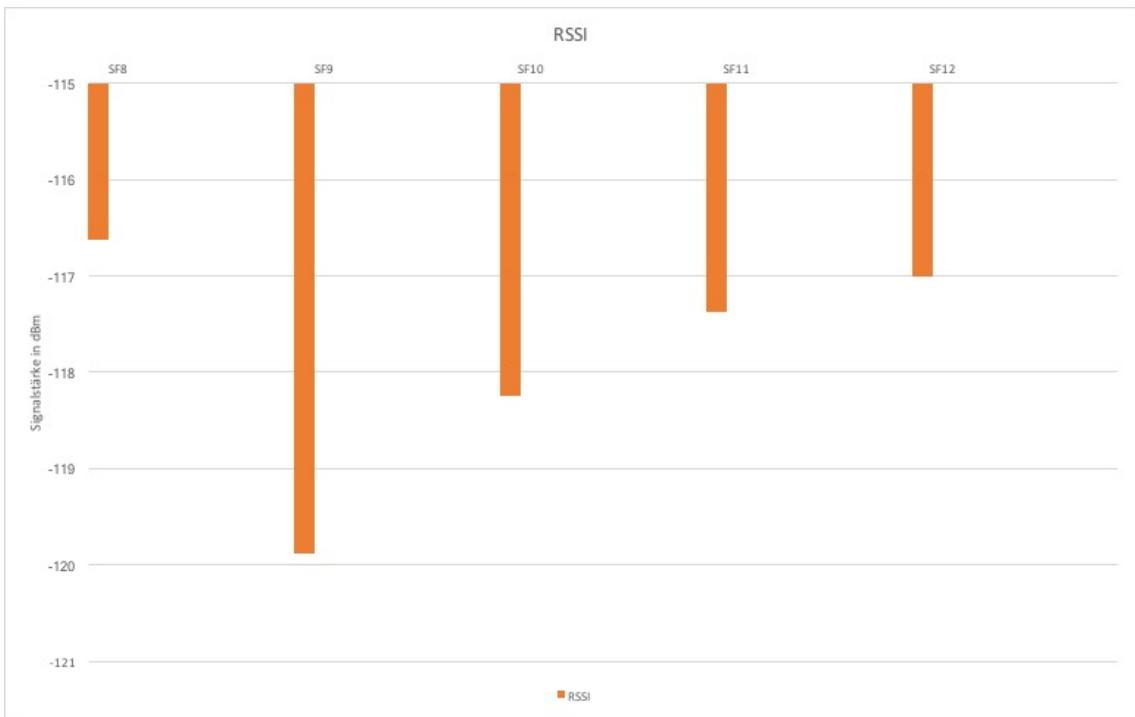


Abbildung 6.9: Messwerte RSSI pro Spreadingfactor

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.

Test 2.2

Ausgangslage: Analog Test 2.1. Die Messungen werden in der Region 2 durchgeführt. Anstatt wie im ersten Test 10 Messungen durchzuführen, wird die Anzahl auf 50 erhöht. Ausreißer werden ebenfalls entfernt. 20% der Messwerte fallen weg. Damit bleiben 40 Messpunkte.

Annahme: Analog Test 2.1

Die Messresultate:

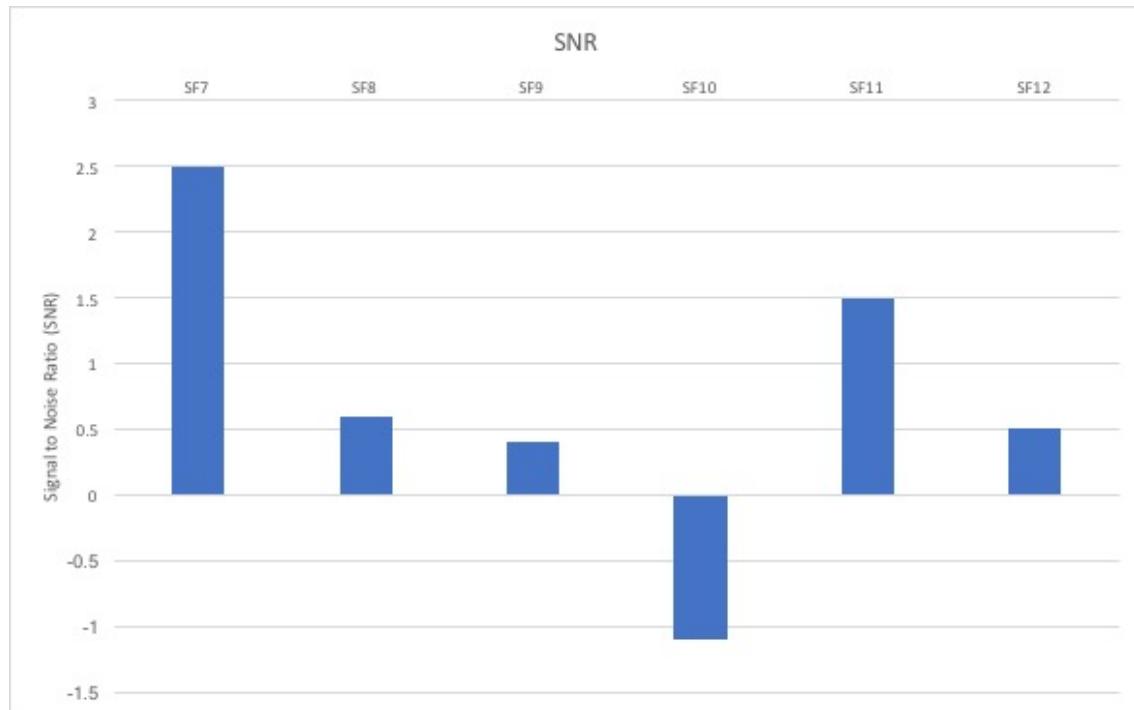


Abbildung 6.10: Messwerte SNR pro Spreadingfactor

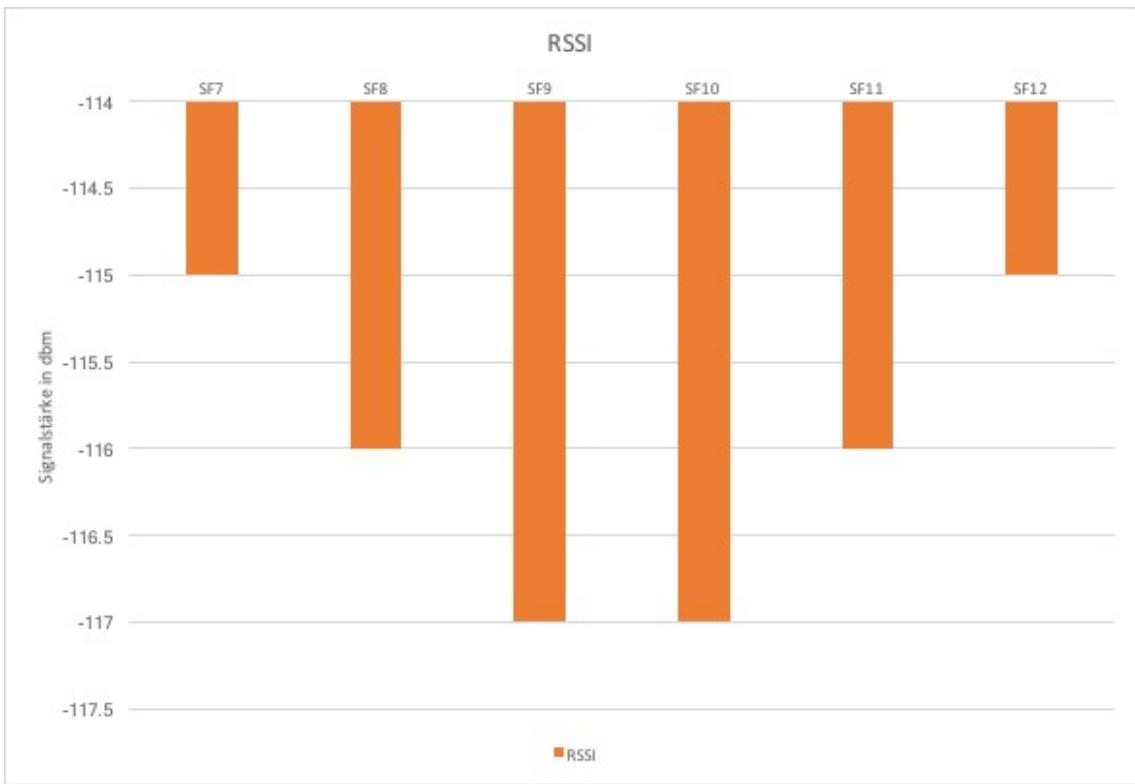


Abbildung 6.11: Messwerte RSSI pro Spreadingfactor

Auch mit mehr Messwerten ist die Annahme nicht eingetroffen. Warum sich die Messwerte von der Literatur distanzieren ist zum jetzigen Zeitpunkt nicht bekannt.

6.5.3 Test 3: Datenrate

Ausgangslage: In diesem Test soll ermittelt werden, wie häufig wir die Daten des Trackers übermitteln können. Die maximal zu übertragende Datengröße wurde bereits in einem vorhergehenden Abschnitt definiert und beträgt **20Bytes**. Die Coding Rate wird auf 4/8 gesetzt.

Vorgehen: Pro Spreadingfactor werden Messungen durchgeführt. Die Zeitdifferenz zwischen den Übertragungen wird ermittelt.

Messresultate:



Abbildung 6.12: Zeit zwischen zwei Paketen

Auch bei SF12 liegt die Zeit zwischen den Übertragungen bei nur 170 Sekunden. Das bedeutet, dass wir ca. alle 3 Minuten den neuen Standort übermitteln können. Beim Wandern wird diese Rate ausreichen. Ob diese Zeit für Skifahrer reicht müssten in weiteren Tests festgestellt werden.

Kapitel 7

Schlusswort

7.1 Verbesserungen bestehendes System

7.1.1 Automatischer Datenaustausch

TODO, Prio 2

7.2 Ausblick

7.2.1 Wireless Kommunikation

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 Unrsachen	6
2.1	Atas Schema Grob	10
3.1	Beispiel Upchirp	16
3.2	Wasserfall Darstellung: Chirp Parameter	17
3.3	Wasserfall Darstellung: Spreadingfactors 7-12	18
3.4	Time on Air	20
3.5	LoRaWan Architektur	22
3.6	OTAA Ablauf	24
3.7	LoRaWan Klasse A	26
3.8	LoRaWan Klasse C	27
3.9	Swisscom LoRaWan Abdeckung	29
3.10	TTN Architektur	30
3.11	TTN Blackbox	31
4.1	Legende - Diagram	32
4.2	Legende - Diagram - graue Box	32
4.3	Systemübersicht Detail	33
4.4	Tracker Detail	34
4.5	Gateway Detail	36
4.6	TTN Detail	37
4.7	TTN Detail	38
4.8	ATAS-Service Diagram	39
4.9	ATAS-Web Diagramm	40
4.10	Startseite	41
4.14	Gefahrenzone	43
4.15	MQTT Struktur	44
5.1	Prototyping Ablauf	46
5.2	Flowchart Prototyp 2	47
5.3	Prototyp 2 Antenne	51
5.4	Prototyp2 Schema	53

5.5	Kommunikationsart Prototyp 2	55
5.6	Prototyp 2	56
5.7	Klassendiagramm	58
5.10	Notfallalarm	60
5.11	Fehlerhafter Bildaufbau Display	61
5.13	Gesamt Energieverbrauch pro Prototyp	63
5.14	Veränderung des Energieverbrauchs zum Ruhezustand	64
6.1	Flowchart Testing	66
6.2	TTN Messages	67
6.3	Gateway	73
6.4	TTN Dashboard Gateway	75
6.5	TTN Dashboard Gateway Registriert	75
6.6	ATAS-Web Menu	76
6.7	ATAS-Web Dashboard	77
6.8	Messwerte SNR pro Spreadingfactor	80
6.9	Messwerte RSSI pro Spreadingfactor	81
6.10	Messwerte SNR pro Spreadingfactor	82
6.11	Messwerte RSSI pro Spreadingfactor	83
6.12	Zeit zwischen zwei Paketen	84

Glossar

AppSKey Mittels AppSKey werden die Nutzdaten separat nochmals verschlüsselt. Nur der Nutzer des Netzwerks kann die Daten lesen das Transportnetzwerk bspw. TTN kann diese nicht einsehen.[?]. 30

ATAS Aplinist Tracking & Alerting System. 3

DevAddr Logische Geräte Adresse, analog einer IP Adresse in einem IP Netzwerk.. 30

Downlink Kommunikation vom Gateway zu Endgerät. 17

LoRA Long Range, Low Power long Range Wireless. 9

LoRaWan Low Power Wide Area Network, MAC Protocl. 9

LoRaWan Endgerät LoRaWan Endgerät. Atas-Node ist ein LoRaWan Endgerät. 30

MQTT Message Queuing Telemetry Transport). 9

NetSKey Wird für die Verschlüsselung der Kommunikation zwischen Endgerät und Netzwerkanbieter bspw. TTN verwendet. Mittels Schlüssel kann die Datenintegrität sichergestellt werden d.H. Manipulierte Nachrichten werden erkannt. 30

TTN The Things Network, LoraWan Netzwerk. 9, 17

Uplink Kommunikation vom Endgerät zum Gateway. 17

Literaturverzeichnis

- [1] <http://85.25.34.248/bergmed/bergmed.php?section=15> 1.2.2018
- [2] <https://www.swisscom.ch/en/about/medien/press-releases/2017/06/20170628-mm-5g-speed.html> 5.1.2018
- [3] <https://de.wikipedia.org/wiki/Mehrwegempfang> 5.1.2018
- [4] https://en.wikipedia.org/wiki/Duty_cycle 5.1.2018
- [5] <https://www.swisscom.ch/en/business/enterprise/themen/connectivity/lpn-post-swisscom.html> 5.1.2018
- [6] <http://lpn.swisscom.ch/e/our-coverage/> 5.1.2018
- [7] <https://arxiv.org/pdf/1607.08011.pdf> 4.1.2018
- [8] <https://github.com/matthijskooijman/arduino-lmic> 212.2018
- [9] <https://www.thethingsnetwork.org/article/setting-up-a-private-handler-connected-to-the-public-community-network> 2.2.2018
- [10] <https://www.thethTNningsnetwork.org/docs/applications/apis.html> 2.1.2018
- [11] <https://www.rs-online.com/designspark/learning-lorawan-basics> 1.1.2018
- [12] <https://www.thethingsnetwork.org/wiki/LoRaWAN/ADR> 1.1.2018
- [13] <http://www.erodocdb.dk/Docs/doc98/official/pdf/REC7003e.pdf> 1.1.2018
- [14] <https://github.com/ttn-zh/ic880a-gateway/wiki> 1.1.2018
- [15] <https://wireless-solutions.de/products/radiomodules/ic880a> 1.2.2018
- [16] <https://www.bergwelten.com/a/die-schoensten-zitate-rund-ums-bergsteigen> 1.1.2018
- [17] <http://www.sac-cas.ch/unterwegs/sicherheit/bergnotfallstatistik.html> 1.2.2018

- [18] <https://www.thethingsnetwork.org/forum/t/antenna-length-for-868-and-433-mhz/5378> 2.1.2018
- [19] <https://en.m.wikipedia.org/wiki/Wavelength> 2.1.2018
- [20] http://www.oegan.at/notfallmedizin/index.php?option=com_content&view=article&id=76:unterkuehlung-als-ueberlebenschance&catid=42&Itemid=118 1.01.2018
- [21] <https://www.thethingsnetwork.org/wiki/LoRaWAN/Security> 27.07.2017
- [22] <https://www.jaguar-network.com/en/news/lorawan-in-a-nutshell-2-internet-of-things-iot>, 27.07.2017
- [23] <https://www.thethingsnetwork.org/wiki/LoRaWAN/ADR> 27.07.2017
- [24] <https://www.autodesk.com/products/eagle> 27.07.2017
- [25] <http://iot-bits.com/documentation/esp32-tutorial-and-example-programs/> 20.12.2017
- [26] <https://dl.espressif.com/doc/esp-idf/latest/get-started/index.html> 27.12.2017
- [27] <https://www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers> 27.12.2017