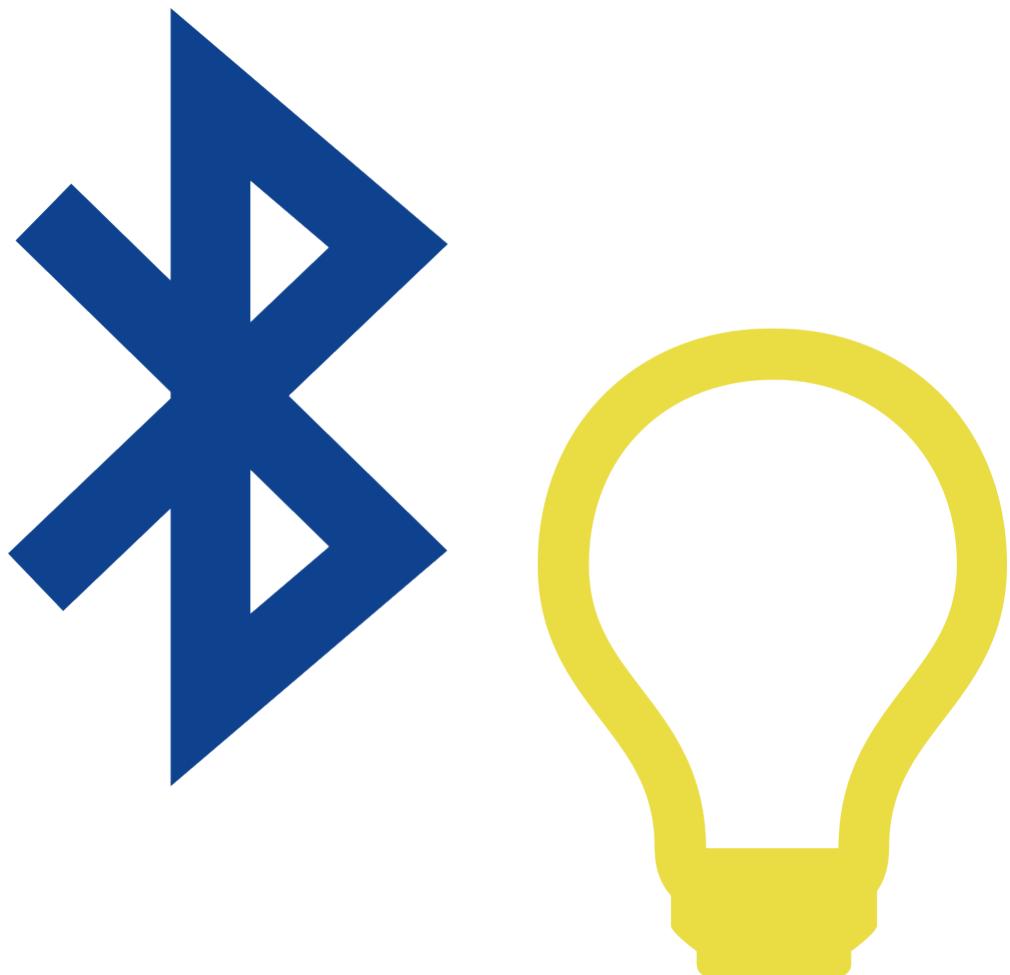


ZHAW Zürcher Hochschule für Angewandte Wissenschaften
Bachelorstudium Informatik
Semesterarbeit
Betreuungsperson: Peter Egli

Heimautomatisierung mit iBeacons



Fabian Vogler, fabian.vogler@gmail.com
18. April 2014, Version 0.1

© 2014 Fabian Vogler
Alle Rechte vorbehalten

Diese Arbeit entstand im Rahmen der Semesterarbeit an der Zürcher Hochschule für Angewandte Wissenschaften (ZHAW) in Zürich.

Dieses Dokument wurde in HTML geschrieben und mit Hilfe von Prince XML in ein PDF-Dokument umgewandelt. Die verwendeten Schriftart ist *Helvetica Neue*, entwickelt von D. Stempel AG und basierend auf *Helvetica* von Max Miedinger.

Inhaltsverzeichnis

1	Einleitung	4
1.1	Ausgangslage	4
1.2	Ziele der Arbeit	4
1.3	Aufgabenstellung	4
1.4	Vorwort	5
1.5	Projektablauf	6
2	iBeacon	7
2.1	Beschreibung	7
2.2	Bluetooth Low Energy	7
2.3	Core Bluetooth	8
2.4	Analyse iBeacon	9
2.5	Estimote Beacon	18
2.6	Messung RSSI	19
3	Anforderungen	27
3.1	Einleitung	27
3.2	Stakeholder	27
3.3	Systemkontext	28
3.4	Funktionale Anforderungen	29
3.5	Nichtfunktionale Anforderungen	36
3.6	Benutzerschnittstellen	38
4	Software	40
4.1	Randbedingungen	40
4.2	Lösungsstrategie	40
4.3	Bausteinsicht	40
4.4	Laufzeitsicht	42
4.5	Verteilungssicht	43
4.6	Datenmodell	44
4.7	Benutzeroberfläche	45
5	Fazit	48
6	Anhang	49
6.1	Software	49
6.2	Quellenverzeichnis	50
6.3	Abbildungsverzeichnis	52
6.4	Glossar	54

1 Einleitung

1.1 Ausgangslage

Heimautomatisierung (engl. Home Automation) bezeichnet die intelligente Verknüpfung von Sensoren und anderen technischen Geräten innerhalb der eigenen Wohnräume zur Steigerung des Wohnkomforts. Effizientere Technik und verbraucherorientierte Produkte haben dies in den letzten Jahren auch für normale Konsumenten realisierbar gemacht.

Apple hat in der neusten Version seines mobilen Betriebssystems, iOS 7, eine neue Möglichkeit zur Lokalisierung des Benutzers in geschlossenen Räumen hinzugefügt. Die Technik basiert auf Bluetooth Low Energy, trägt den Namen iBeacon und kann verwendet werden um das Betreten oder Verlassen eines Raums zu registrieren. Eine Dokumentation des Protokolls wurde von Apple angekündigt jedoch noch nicht veröffentlicht.

Die Verwendung von bestehenden Geräten, wie Smartphones, ist für die Heimautomatisierung interessant, da diese vielfältige Möglichkeiten bieten und nicht separat angeschafft werden müssen. Eine optimale Ergänzung dazu stellt die Philips hue dar. Es handelt sich dabei um LED-Lampe in Form einer Glühbirne, welche über eine API verfügt und so gezielt gesteuert werden kann.

1.2 Ziele der Arbeit

Im Rahmen dieser Semesterarbeit soll analysiert werden, wie die neuen Funktionen von iOS 7 optimal verwendet werden können, um eine einfache Heimautomatisierung zusammen mit der Philips hue zu ermöglichen. Es soll eine zentrale Web-Applikation entwickelt werden, welche die Bewegungen eines Benutzers innerhalb der eigenen Wohnräume speichert und die Beleuchtung der Räume entsprechend steuert. Der Standort des Benutzers wird dabei von einer iPhone-App ermittelt und an die Web-Applikation gemeldet.

1.3 Aufgabenstellung

Im Rahmen dieser Semesterarbeit werden vom Studenten folgende Aufgaben ausgeführt:

1. Analyse der Funktionalitäten von iBeacon.
2. Untersuchung des auf Bluetooth Low Energy basierenden Protokolls von iBeacon.
3. Anforderungsdokumentation für die Automatisierung der Beleuchtung einer Wohnung.

4. Konzeption und Design des Software-Prototyps zur Umsetzung der dokumentierten Anforderungen.
5. Implementation eines Prototyps bestehend aus einer iPhone-App zur Lokalisierung des Benutzers und einer Web-Applikation zur Überwachung und Steuerung der Beleuchtung.
6. Verifikation des Software-Prototyps in einem Feldversuch und Analyse der Ergebnisse.
7. Demonstration des Software-Prototyps.

1.4 Vorwort

Es besteht die Vermutung, dass die Ortung mit iBeacons zu ungenau ist. Mit einem Gerät, welches Bluetooth verwendet, um einen Alarm auszulösen, sobald man einen sich vom iPhone entfernt, wurde beobachtet, dass dies eher unzuverlässig funktioniert. Ohne eine detaillierte Analyse oder Kenntnisse über die genau Funktionsweise ist es jedoch schwierig zu beurteilen, wo die Problemursache liegt.

Zu Beginn des Projekts gab es von Apple keine offizielle Dokumentation zum iBeacon-Format und auch am Ende des Projekts war diese nur zertifizierten Herstellern zugänglich.

Die Hue App selbst bietet bereits eine Steuerung der Beleuchtung über Geo-Location (Wifi, GPS) an. Einige Versuche zeigten jedoch, dass dies eher schlecht funktioniert, da die Ortung besonders im Gebäude nicht zuverlässig funktioniert (Lichter gehen an und aus während sich die Person am gleichen Ort aufhält).

1.5 Projektablauf

Das folgende Gantt-Diagramm gibt einen Überblick über den effektiven Ablauf des Projekts und dessen wichtigsten Meilensteine.

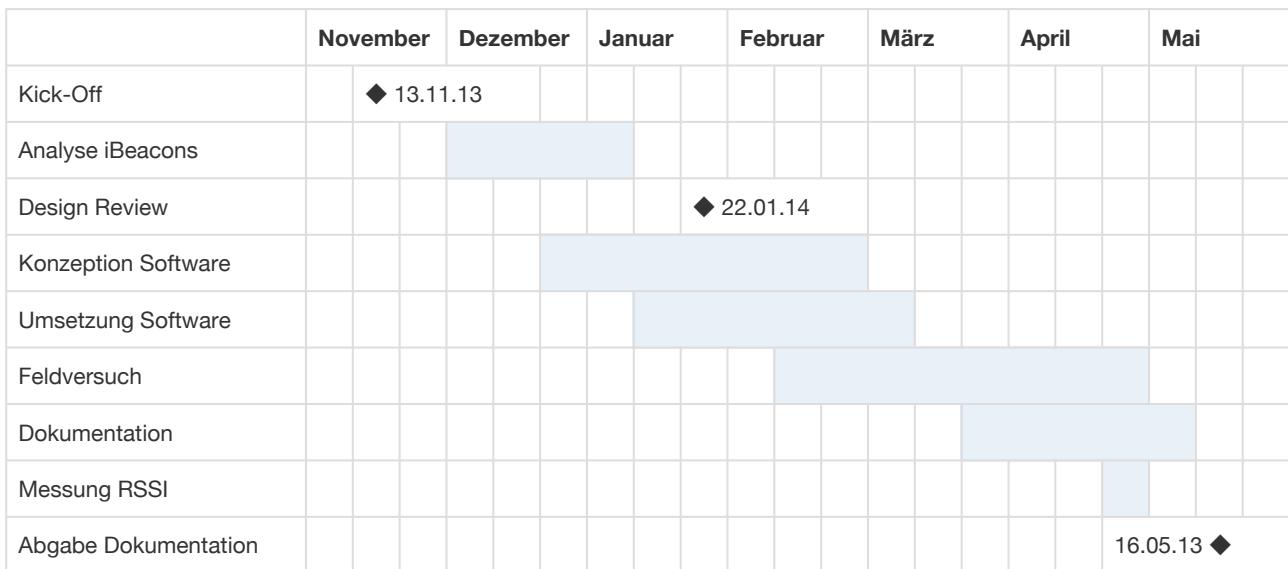


Abb. 1 Projektablauf

Laut dem Reglement für Semesterarbeiten sollte der Aufwand 120 Stunden betragen. Entsprechend wurde die Planung darauf ausgelegt. Die genauen technischen Anforderungen wurden allerdings erst während dem Projekt klar und führten zu einer iterativen Softwareentwicklung, wodurch mehr Zeit für die Konzeptions- und Umsetzungsphasen gebraucht wurde.

Beschreibung	Soll	Ist
Analyse iBeacons	10 h	6 h
Konzeption Software	10 h	14 h
Umsetzung Web-Applikation	25 h	19 h
Umsetzung Automatische Steuerung	15 h	35 h
Umsetzung App	25 h	12 h
Messung RSSI	-	23 h
Dokumentation schreiben	35 h	39 h
Total	120 h	147 h

Abb. 2 Soll / Ist Vergleich Aufwand

2 iBeacon

2.1 Beschreibung

Bei iBeacon handelt es sich um eine von Apple eingetragene Marke und beschreibt einen proprietären Technologie für die Lokalisierung eines iPhones mithilfe von Bluetooth Low Energy (BLE). Es setzt das Betriebssystem iOS 7 oder neuer, sowie ein iPhone mit Bluetooth 4.0 voraus. Das Wort Beacon wird zur Beschreibung eines Geräts mit eigener Stromversorgung verwendet, welches einen Standort markiert.



Abb. 3 iBeacon Logo [1]

Kommt ein iPhone in die Nähe eines Beacons, wird eine auf dem iPhone laufende App vom Betriebssystem benachrichtigt. Die App kann dann zusätzlich die Distanz zum Beacon abfragen und eigene Aktionen ausführen. [2]

2.2 Bluetooth Low Energy

Bluetooth Low Energy (BLE) ist ein Funkprotokoll und Teil der Bluetooth 4.0-Spezifikation. Vermarktet wird BLE auch als Bluetooth Smart. Es wurde speziell für kleine Geräte mit limitierten Akkukapazitäten entwickelt und kommt deshalb oft auf Smartphones zum Einsatz. [3]

Bluetooth Low Energy-Geräte nutzen das Generic Attribute Profile (GATT), um ihre Funktionalität zu beschreiben. GATT sieht Services und Characteristics, welche ein Gerät im Rahmen eines Profils anbieten kann. Dabei besteht eine hierarchische Beziehung zwischen Services und Characteristics. Profile dienen zur formellen Gruppierung von mehreren Services.

[1] Estimote, Inc. (2013): Estimote Beacons

[2] Apple Inc. (2013): iOS: Understanding iBeacon

[3] Bluetooth SIG, Inc. (2013): Bluetooth Smart

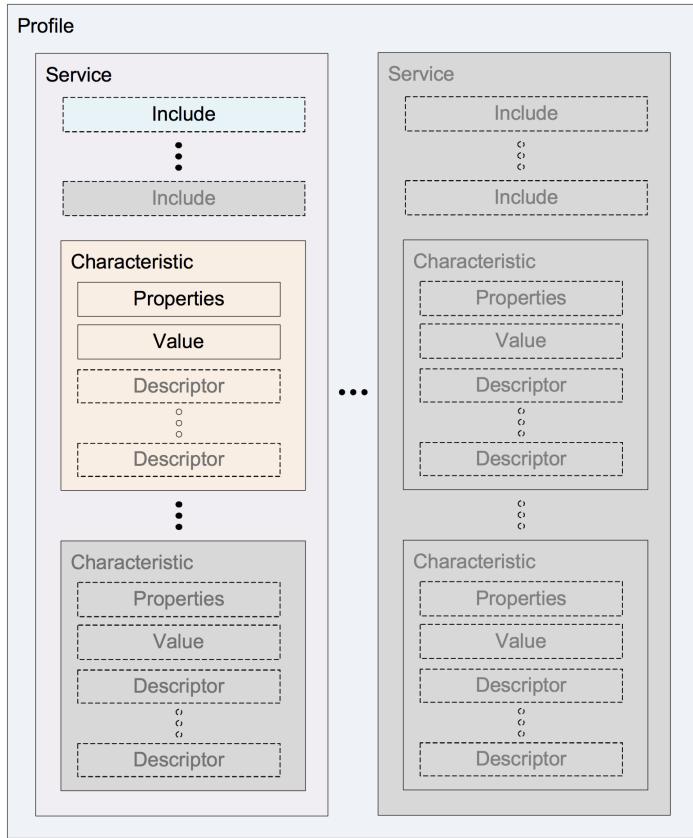


Abb. 4 GATT Profilhierarchy [4]

2.3 Core Bluetooth

Apple bietet mit Core Bluetooth eine vereinfachte Schnittstelle zur Verwendung von BLE an, welche die BLE-Protokolle abstrahiert. [5]

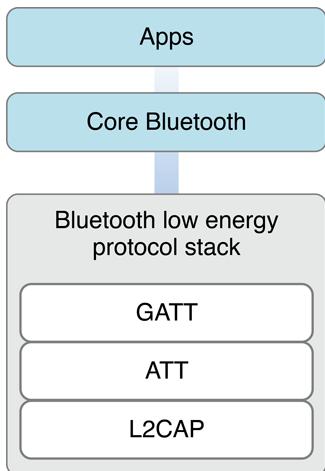


Abb. 5 Aufbau Core Bluetooth [6]

[4] Seite 532, 2.6 GATT Profile Hierarchy, Generic Attribute Profile (GATT), Vol 3, Bluetooth SIG, Inc. (2010): Bluetooth Specification Version 4.0

[5] Apple Inc. (2013): Core Bluetooth Programming Guide

[6] Seite 5, About Core Bluetooth, Apple Inc. (2013): Core Bluetooth Programming Guide.

Dabei wird bei der Programmierung zwischen zwei Rollen welche nach dem Client-Server-Prinzip aufgebaut sind unterschieden: Central und Peripheral. Ein «Peripheral» (Server) stellt dabei Daten zur Verfügung und kann sich selbst über das regelmässige Aussenden von Informationen ankündigen (*Advertising*). Ein «Central» (Client) hingegen sucht nach Peripherals, und verarbeitet dessen Informationen in weiteren Aktionen. Zum Beispiel kann ein digitaler Pulsmesser (Peripheral) mit BLE die aktuelle Herzfrequenz an eine iOS-App (Central) übermitteln und die iOS-App zeigt dann diese dem Benutzer an.

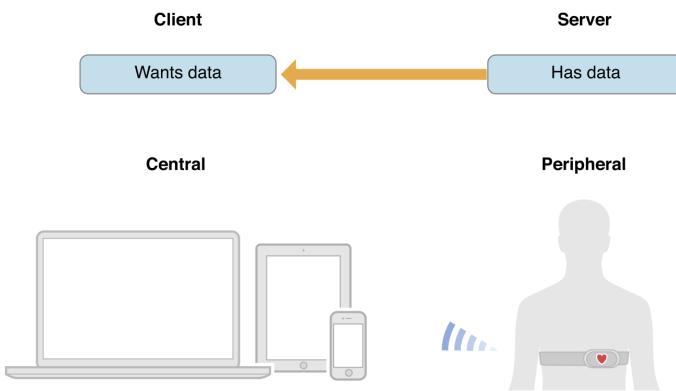


Abb. 6 Rollen in Bluetooth Low Energy [7]

2.4 Analyse iBeacon

Um der Funktionsweise von iBeacon auf die Spur zu kommen, wurde mit der Einarbeitung in die Grundtechnologie, auf der iBeacon basiert begonnen. Mit dem indirekten Ziel, einen Computer als iBeacon zu nutzen, wurde mit der Analyse auf einem Mac begonnen. Dazu wurde das bereits beschrieben Core Bluetooth Bibliothek verwendet, welche seit Version 10.9 von Mac OS X nicht nur auf iOS zur Verfügung steht. Damit sollte ein einfaches BLE-Profile veröffentlicht werden.

Anhand des Kapitels «Performing Common Peripheral Role Tasks» (Seite 16) aus der von Apple zur Verfügung gestellten Dokumentation «Core Bluetooth Programming Guide [8]» wurde ein minimales Programm mithilfe der Programmierumgebung Xcode erstellt, welches eigene BLE-Dienste anbietet.

```
// create manager
CBPeripheralManager *manager = [[CBPeripheralManager alloc] initWithDelegate:self queue:nil
options:nil];

// initialize UUIDs
CBUUID *characteristicUUID = [CBUUID UUIDWithString: @"2A38"];
CBUUID *serviceUUID = [CBUUID UUIDWithString: @"180D"];
```

[7] Seite 9, Central and peripheral devices, Apple Inc. (2013): Core Bluetooth Programming Guide.

[8] Apple Inc. (2013): Core Bluetooth Programming Guide

```
// create value object
int heartRate = 42;
NSData *value = [NSData dataWithBytes: &heartRate length: sizeof(heartRate)];

// create service with characteristic
CBMutableCharacteristic *characteristic = [[CBMutableCharacteristic alloc]
initWithType:characteristicUUID properties:CBCharacteristicPropertyRead value:value
permissions:CBAtributePermissionsReadable];
CBMutableService *service = [[CBMutableService alloc] initWithType:serviceUUID primary:YES];
service.characteristics = @[characteristic];

// add service and advertise
[manager addService:service];
[manager startAdvertising:@{ CBAdvertisementDataServiceUUIDsKey : @[[serviceUUID] }];
```

Abb. 7 Beispielanwendung von Core Bluetooth

Konkret wurde mit der Beispiel-Anwendung mit einem Service für Herzfrequenzen mit dem UUID **180D** und mit einer Charakteristik für die Position des Sensors am Körper mit der UUID **2A38** entwickelt.

Die iPhone-App **LightBlue** [9] bietet die Möglichkeit, eine Verbindung zu beliebigen BLE-Geräte aufzubauen und die von den Geräten angebotenen Dienste abzurufen. Die App zeigt zudem die vom Gerät versendeten Advertising-Daten an. Die App wurde erfolgreich dazu verwendet, die von der Mac-Anwendung publizierten Services und Characteristics zu finden.

[9] Punch Through Design LLC (2013): LightBlue - Bluetooth Low Energy

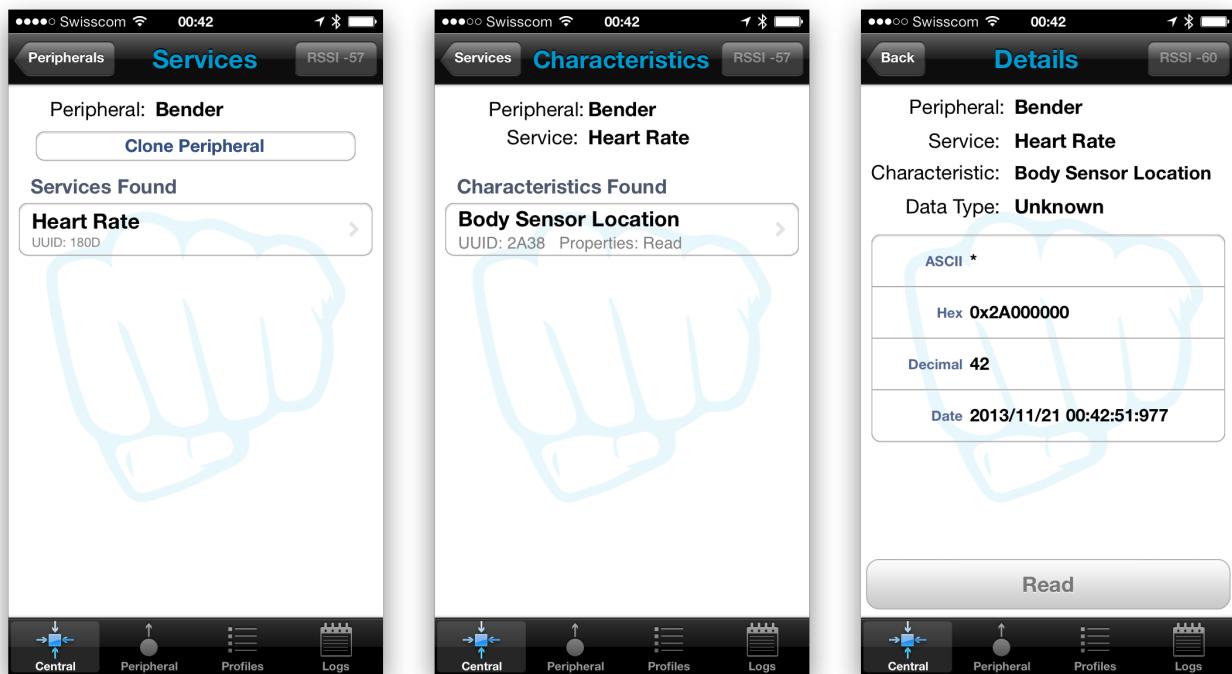


Abb. 8 Screenshot LightBlue

Wie im Kapitel Turn Your iOS Device Into a Beacon [10] der Dokumentation zu den iBeacons beschrieben, wurde eine einfach iPhone-App erstellt, welche die Signale eines iBeacons vom iPhone aussendet. Dazu benötigt werden die beiden Bibliotheken *CoreBluetooth.framework* und *CoreLocation.framework*.

Jedes Beacon braucht eine eindeutige UUID, anhand welcher es identifiziert wird. Um die Analyse des Protokols zu vereinfachen und die ID einfach auffindbar wurde die UUID FFFFFFFF-FFFF-FFFF-FFFF-FFFFFFFFFFF definiert. Als Major wurde der Wert 00 (0) verwendet und als Minor FF (65535).

```
NSUUID *proximityUUID = [[NSUUID alloc]
initWithUUIDString:@"FFFFFFFF-FFFF-FFFF-FFFF-FFFFFFFFFFF"];

// Create the beacon region.
CLBeaconRegion *beaconRegion = [[CLBeaconRegion alloc] initWithProximityUUID:proximityUUID
major:0 minor:65535 identifier:@"ch.zhaw.voglefab.debug"];

// Create a dictionary of advertisement data.
NSDictionary *beaconPeripheralData = [beaconRegion peripheralDataWithMeasuredPower:nil];

// Start advertising your beacon's data.
[self.peripheralManager startAdvertising:beaconPeripheralData];
```

Abb. 9 Beispieldaten von Core Bluetooth

[10] Apple Inc. (2013): Location and Maps Programming Guide

Diese App wurde auf einem zweiten iPhone installiert und gestartet. Verbindet man sich jedoch nun mit LightBlue auf dem ersten iPhone mit dem zweiten iPhone, werden in LightBlue nur die Services *Battery Service* und *Current Time Service* angezeigt - iBeacon wird also kein Service im GATT-Protokoll. Auch bei den Advertisement-Informationen ist das iBeacon in LightBlue nicht sichtbar. Trotz der neuen Erkenntnis stellt sich dieser Lösungsansatz somit aber als Sackgasse heraus.

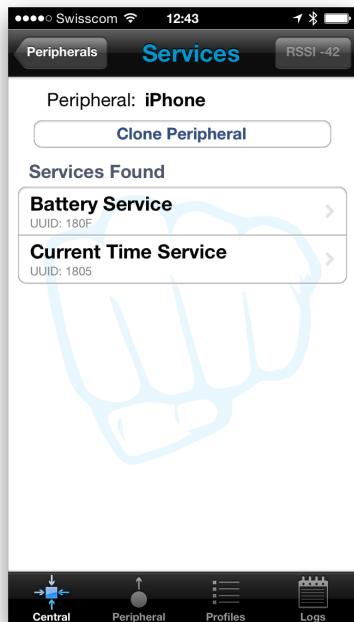


Abb. 10 Screenshot LightBlue iBeacon

In einem weiteren Schritt wurde auch nach Analyse-Tools auf dem Mac gesucht. Schliesslich wurde das Software-Paket Hardware IO Tools for Xcode [11] gefunden, welches von Apple für Entwickler gratis zum Download bereit gestellt wird. Dieses enthält unter anderem die Tools *PacketLogger* und *Bluetooth Explorer*, welche für die Analyse der iBeacons verwendet wurden.

[11] Apple Inc. (2013): Hardware IO Tools for Xcode

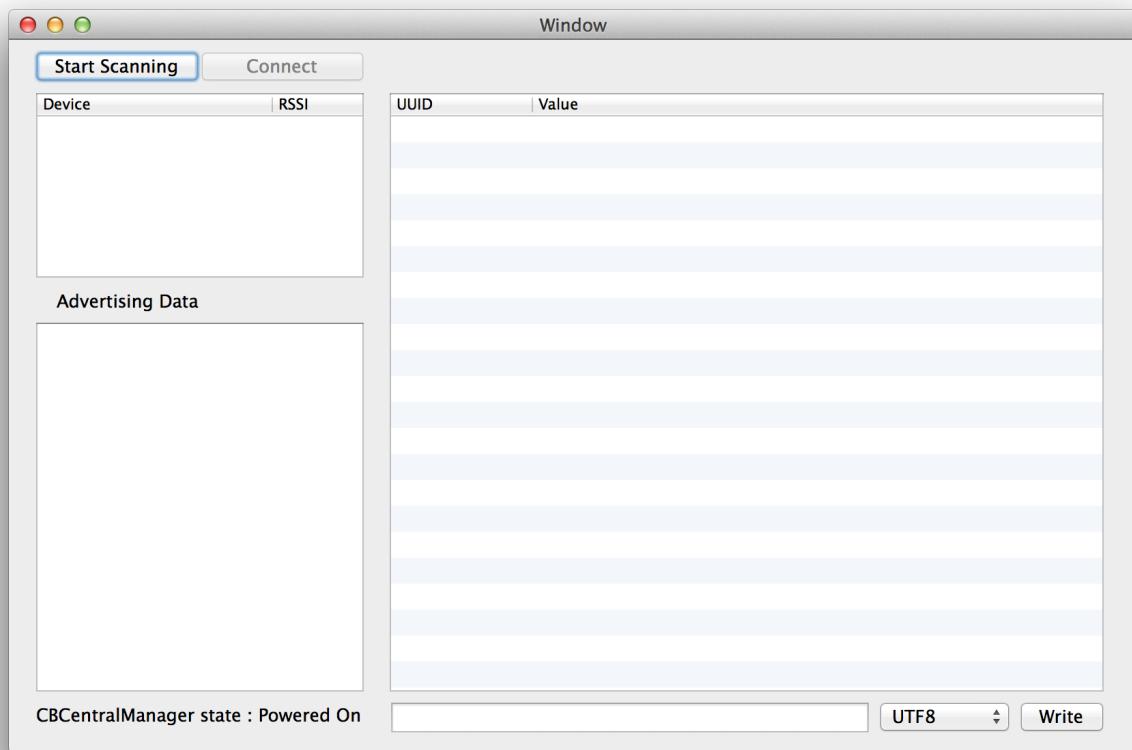


Abb. 11 Screenshot Bluetooth Explorer

Das neue Ziel war nun, die Signale eines iBeacons auf dem Mac zu analysieren. Die Software PacketLogger bietet dazu eine Möglichkeit, die vom Computer empfangenen BLE-Packete zu analysieren. Scan man mit dem Bluetooth Explorer nach BLE-Geräten, werden im PacketLogger die empfangenen BLE Advertisements angezeigt.

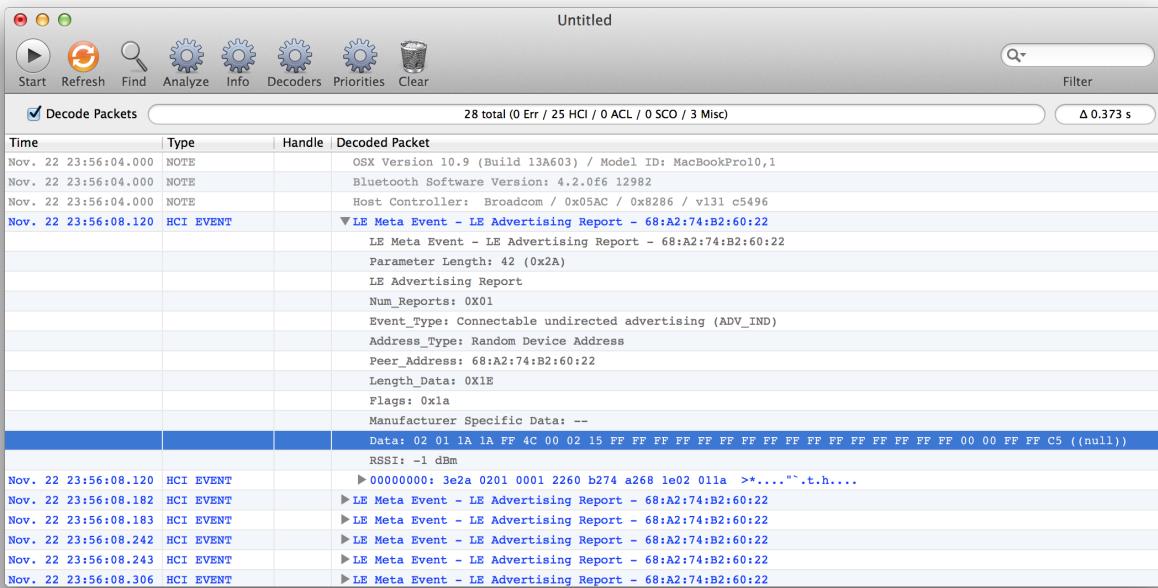


Abb. 12 Screenshot PacketLogger

Dadurch wurde schlussendlich das Advertisement von der iPhone-App mit den Daten gefunden, welche das iBeacon-Signal aussendet. Verwendet wird hier die hexadezimale Schreibweise von Octet, zwei Hexzahlen entsprechen also einem Byte resp. 8 Bits.

Abb. 13 Advertisement Daten iBeacon

Der Aufbau des Advertisements wird auf Kapitel 11 *Advertising and Scan Response data format* im Volume 3 der Bluetooth-Spezifikation [12] erklärt. Ein Advertisement besteht aus mehreren Elementen, welche eine variable Länge haben. Am Anfang jedes Elements steht seine eigene Länge gefolgt vom Typ des Elements.

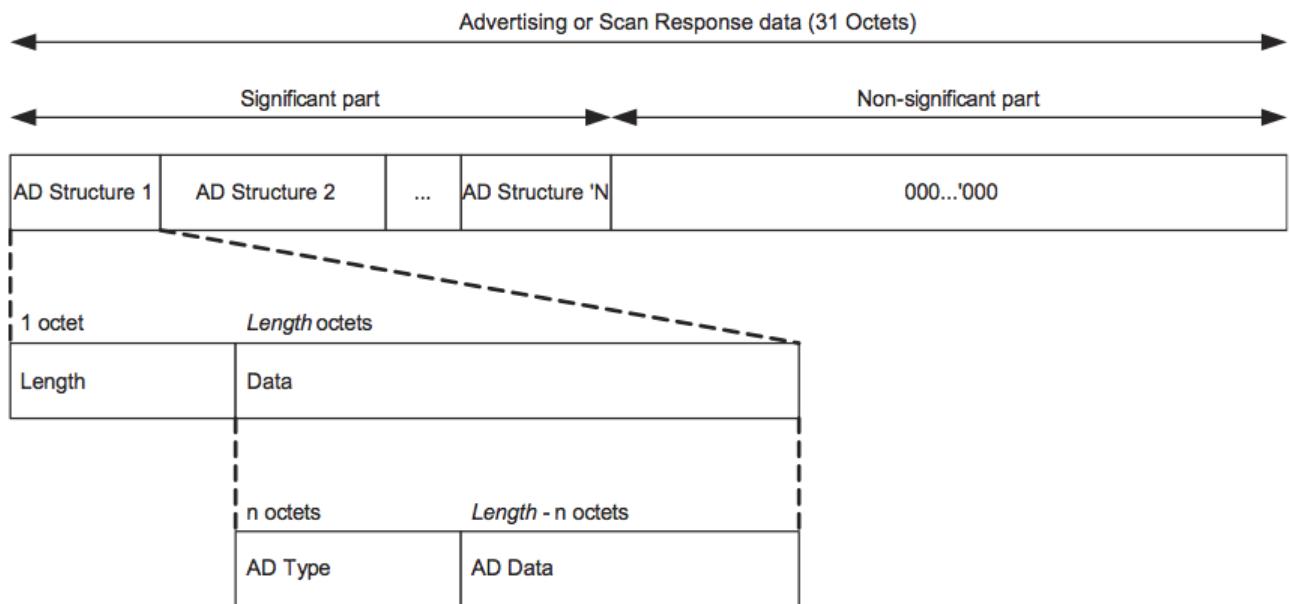


Abb. 14 Struktur Advertising and Scan Response [13]

Angewendet auf das empfangene Advertisement ergibt sich folgendes Format für ein iBeacon. Gemäss Spezifikation enthalten die ersten zwei Octets des Manufacturer Specific Data den Company Identifier Code, welcher auf der Website [14] der Bluetooth SIG dokumentiert ist.

[13] Seite 375, Bluetooth SIG, Inc. (2010): Bluetooth Specification Version 4.0

[14] Bluetooth SIG, Inc. (2014): Company Identifiers

Byte	Beschreibung	Wert
02	Length	2
01	AD Type	Flags
1A	Flags	LE General Discoverable Mode Simultaneous LE and BR/EDR (Controller) Simultaneous LE and BR/EDR (Host)
1A	Length	26
FF	AD Type	Manufacturer Specific Data
4C	Company Identifier Code	Apple, Inc.
00		
02	Fixer Wert	iBeacon Identifier
15		
FF	iBeacon UUID	FFFFFFFF-FFFF-FFFF-FFFF-FFFFFFFFFFFF
FF		
00	iBeacon Major	0
00		
FF	iBeacon Minor	65535
FF		
C5	Fixer Wert	Measured power

Abb. 15 Bluetooth Advertisement iBeacon

Das iBeacon-Format ist also innerhalb der Manufacturer Specific Data im BLE-Protokoll gekapselt. Dies konnte auch mit dem Bluetooth Explorer verifiziert werden, der ebenfalls die Manufacturer Data anzeigt. Um ein iBeacon zu erkennen oder zu simulieren, braucht man also legendlich ein Advertisement mit den entsprechenden Manufacturer Specific Data zu senden/empfangen.

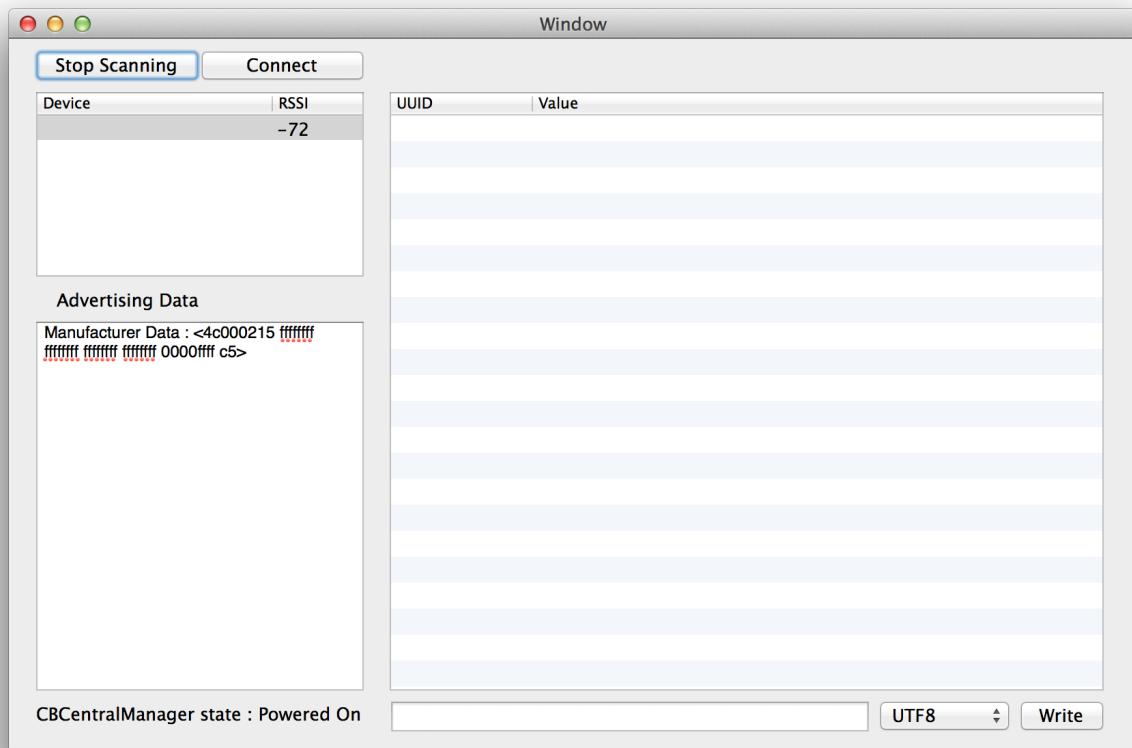


Abb. 16 Screenshot Bluetooth Explorer nach Scan

2.5 Estimote Beacon

Für das Projekt wurden iBeacons von Estimote eingesetzt. Die «Estimote Beacons» wurden hauptsächlich für den Einsatz in Läden entwickelt und enthalten einen Bluetooth-Sender, der das iBeacon-Signal sendet. Da iBeacon eine neue Technologie ist, existierten zu Beginn des Projekts nur wenige Anbieter, welche Hardware für iBeacons anboten. Die Estimote Beacons wurde aufgrund ihrer Verfügbarkeit gewählt.

Estimote verfügt über eine eigene iPhone-App, mit welcher sich die Signalstärke und der Sendeintervall der Beacons konfigurieren lässt. Die Beacons werden eigentlich wasserfest produziert. Leider waren die Batterien nach 5 Monaten aber bereits leer. Per Mail erhielt ich dann vom Estimote-Support eine Anleitung zum Austausch der Batterien. Für den Austausch muss das Beacon mit einem Messer aufgeschnitten werden - die Wasserfestigkeit geht dadurch natürlich verloren.

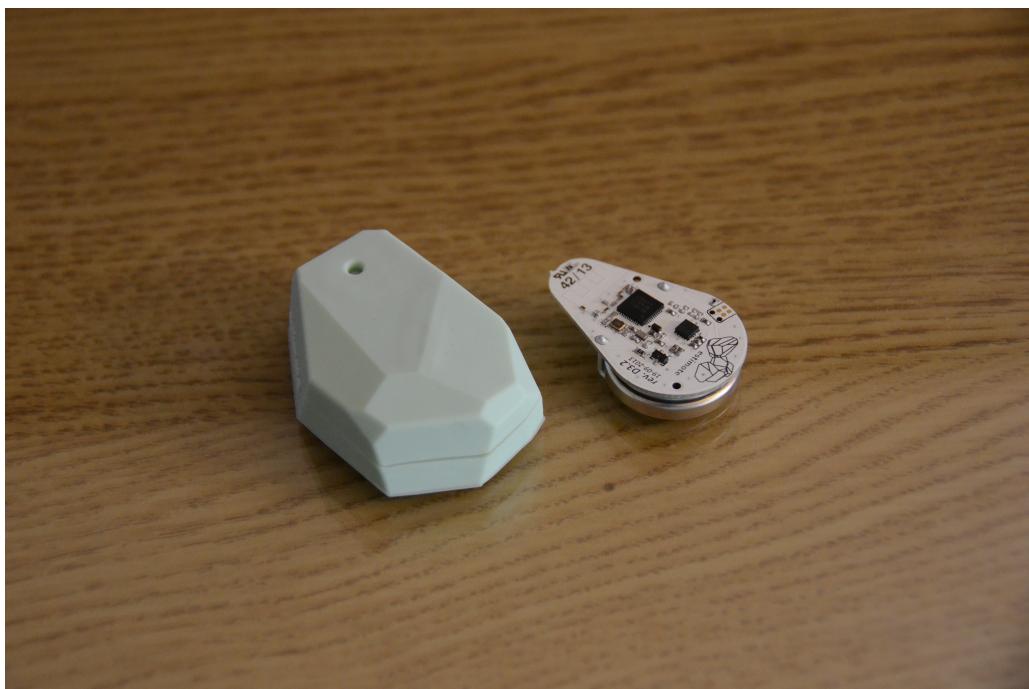


Abb. 17 Foto Estimote Beacon

Der Austausch der Batterie war dennoch interessant, da ich dadurch einen Blick in den Inhalt der Beacons erhalten konnte. Die Beacons bestehen aus einer Leiterplatte mit Schaltkreisen. Die Batterie selbst ist eine Knopfbatterie vom Typ CR2450 und kann im Detailhandel gekauft werden.

2.6 Messung RSSI

Da sich die Signalstärke in den Estimote Beacons konfigurieren lässt, stellt sich die Frage, wie die optimale Konfiguration innerhalb einer Wohnung aussieht. Für folgende Wohnung wurden mehrere Messungen durchgeführt, um für jedes Beacon eine Signalstärke zu finden, die den Anforderungen entspricht.

Die empfangene Signalstärke kann in iOS über den RSSI-Wert (Received Signal Strength Indicator) ausgelesen werden. Je grösser der Wert ist, umso besser ist der Empfang, resp. umso näher befindet sich der Empfänger am Sender.

In einem ersten Schritt wurden die Bauzeichnungen der Wohnung eingescannt und mit der Gratis-Software Sweet Home 3D [15] digitalisiert. Die Wohnung besteht aus zwei Stockwerken, welche mit einer Treppe verbunden sind. Auf dem ersten Stock befindet sich der Wohn- und Bürobereich.

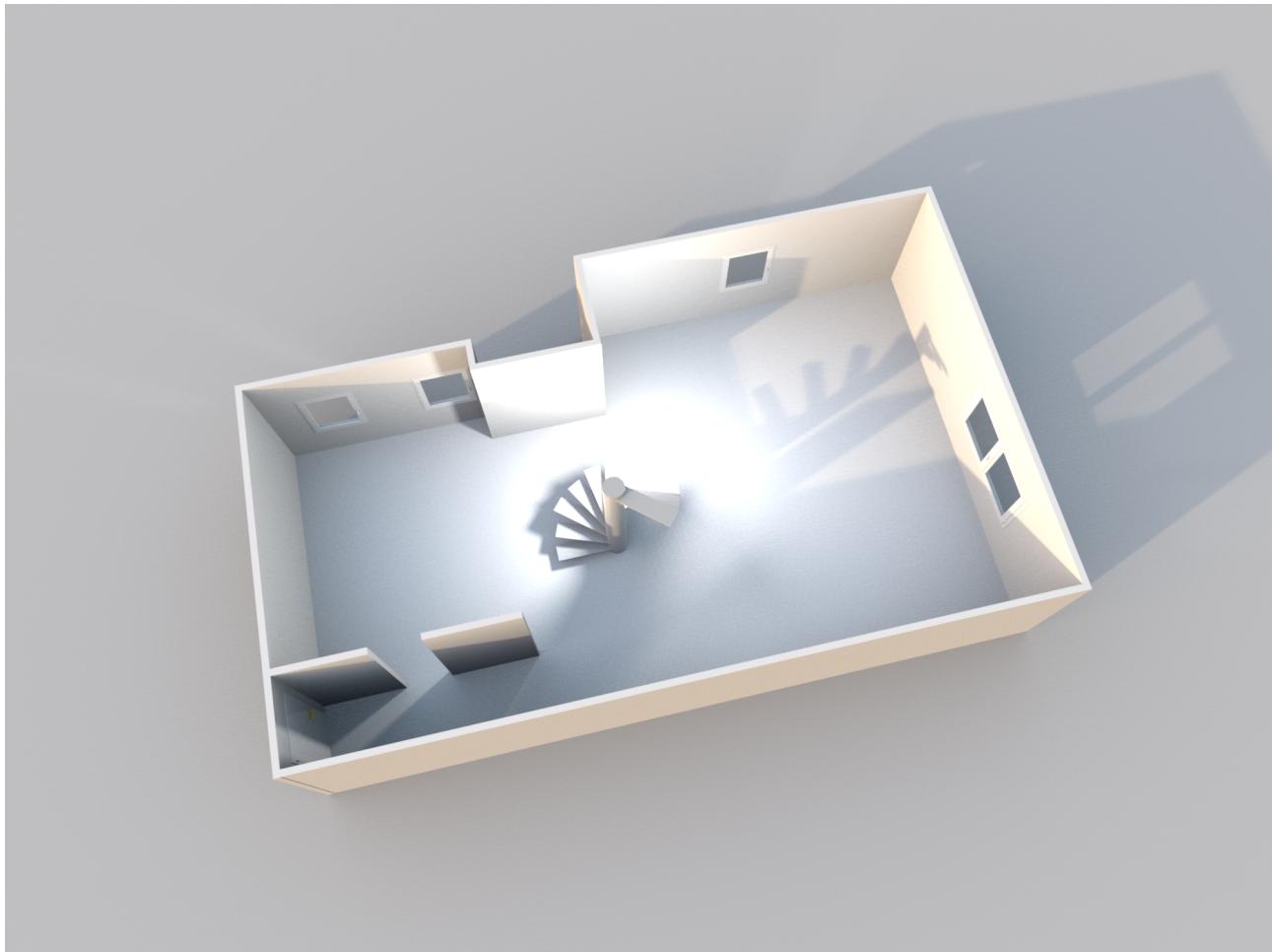


Abb. 18 Visualisierung erster Stock

[15] eTeks (2014): Sweet Home 3D

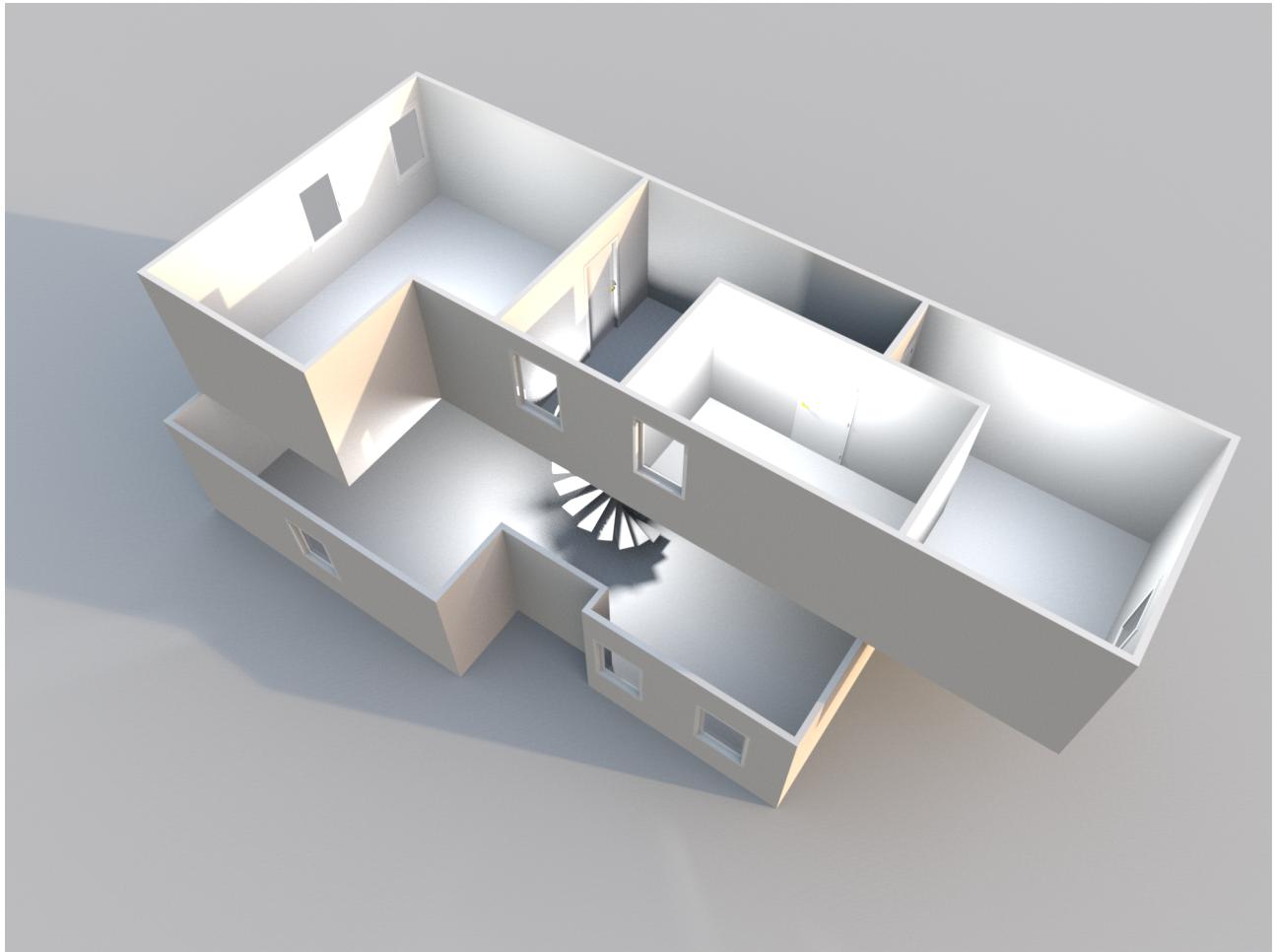


Abb. 19 Visualisierung zweiter Stock

Im zweiten Stock finden sich zwei Schlaf- sowie ein Badezimmer. Das Badezimmer befindet sich in der Visualisierung auf der rechten Seite.

In einem nächsten Schritt wurde die Wohnung in Raster unterteilt. Jedes Quadrat hat im Raster eine Seitenlänge von 120 cm. Diese Länge wurde aufgrund der Gangbreite von 120 cm gewählt, da der Gang normalerweise in der Mitte durchschritten wird und eine mehrfache Messung dort nur wenig Sinn machen würde. Das Raster wurde mit Malerklebband auf die Wohnung übertragen, um die Durchführung der Messung effizient zu gestalten.

Für die Messung wurde eine iPhone-App entwickelt, auf welcher sich die Signalstärke sowie die Koordinaten des iPhones innerhalb der Wohnung konfigurieren lassen. Die App empfängt alle iBeacon-Signale und deren RSSI-Wert. Aktiviert man dann die Messung, wartet die App 10 Sekunden und überträgt dann die empfangenen RSSI-Werte an einen Web-Server zur Auswertung.



Abb. 20 Screenshot iPhone-App

Gemessen wurde auf einer Höhe von 90 cm. Auf dieser Höhe befindet sich das iPhone normalerweise auch, wenn es in der Hosentasche getragen wird. Für die Messung wurde das iPhone auf einem Stativ montiert, damit die Messung ohne Störungsquellen durchgeführt werden kann.



Abb. 21 Fotos Messung

Für die Messung wurde das Advertising Interval alle Estimote Beacons auf 500 ms eingestellt. Das iPhone hatte zu Beginn einer Messung jeweils 100% Akku. Der Batteriestand fiel während der Messung nie unter 60%. Auch die Estimote Beacons hatten frische Batterien.

Während eines ersten Probelauf mit 10 Sekunden warten und 30 Sekunden messen zeigte sich, dass die 10 Sekunden Wartezeit genügen, um sich ausserhalb des Messbereichs zu begeben. Die 30 Sekunden Messzeit erwiesen sich allerdings als zu lang, anhand der aufgezeichneten Daten wurde deshalb die Messzeit auf 20 Sekunden reduziert.

Ablauf einer Messung:

1. Zu messender Signalstärke definieren
2. Alle iBeacons auf die definierte Signalstärke einstellen
3. App auf iPhone starten
4. Signalstärke in App eintragen
5. iPhone auf Stativ montieren
6. Stativ in der Mitte eines Felds platzieren
7. Koordinaten von Plan ablesen und in App eintragen
8. Schalter "Messung" in App auf aktiv stellen
9. Sich innerhalb von 10 Sekunden aus dem Messfeld entfernen
10. 30 Sekunden warten
11. Zurück zu Schritt 6 bis alle Felder/Koordinaten gemessen wurden

Messprotokolle

Die folgenden Messprotokolle geben Auskunft über die Durchführung der Messungen.

Signalstärke	-12 dBm
Datum	01.05.2014
Zeit Beginn	10:46
Zeit Ende	12:40
Messende Person	Fabian Vogler
Notizen	Beim Start der Messung des 2. Stockwerks wurde festgestellt, dass ein Beacon nicht gemessen wurde. Offenbar hat das Beacon kurzzeitig keine Daten gesendet. Der Wert wurde gelöscht und die Messung für dieses Feld erfolgreich wiederholt.

Abb. 22 Protokoll Messung 1

Signalstärke	-4 dBm
Datum	01.05.2014
Zeit Beginn	16:39
Zeit Ende	17:58
Messende Person	Fabian Vogler
Notizen	Es sind keine Probleme aufgetreten.

Abb. 23 Protokoll Messung 2

Signalstärke	-20 dBm
Datum	06.05.2014
Zeit Beginn	09:12
Zeit Ende	10:31
Messende Person	Fabian Vogler
Notizen	Neu wurde ein iPad zur laufenden Überwachung der Resultate eingesetzt. Es sind jedoch keine Probleme festgestellt worden.

Abb. 24 Protokoll Messung 3

Bei den nachfolgenden Auswertungen wurde die Position des gemessenen Beacons im Wohnungsplan jeweils durch eine schwarze Raute gekennzeichnet.

Auswertung Beacon 1

Dieses Beacon befindet sich im Wohnzimmer der Wohnung. Die Signalstärke -12 dBm wird aufgrund der guten Abdeckung des ganzen ersten Stocks für dieses Beacon festgelegt.

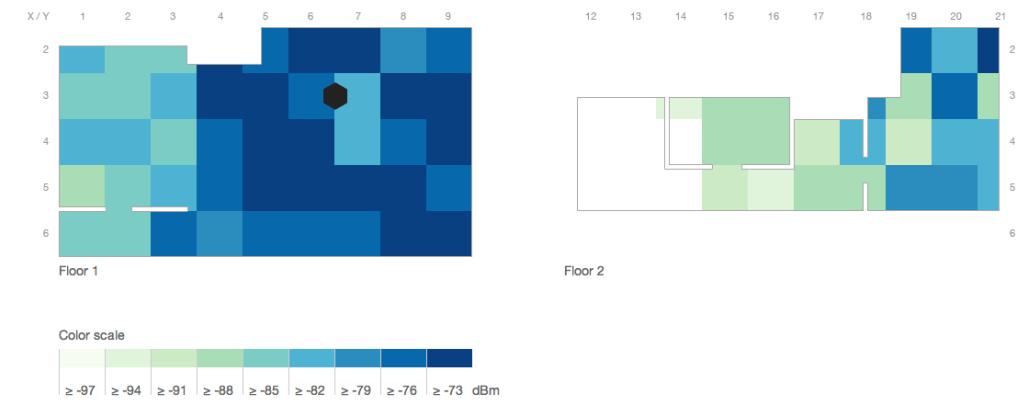


Abb. 25 Visualisierung RSSI Beacon 1, -4 dBm

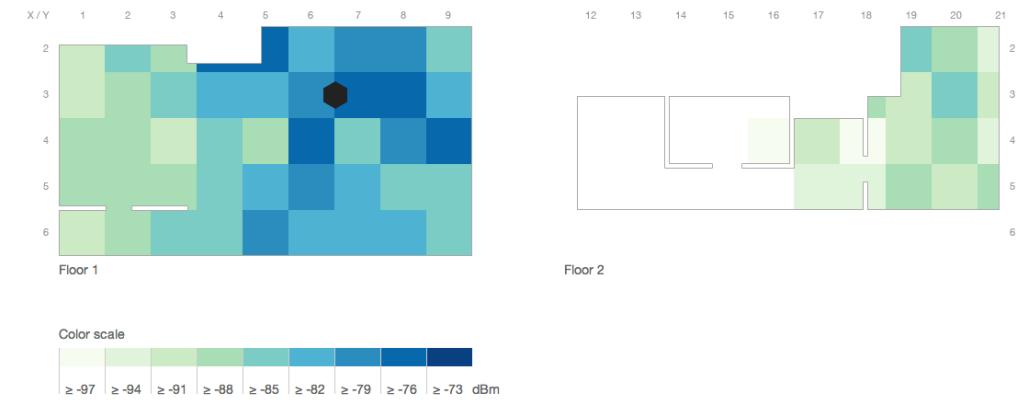


Abb. 26 Visualisierung RSSI Beacon 1, -12 dBm

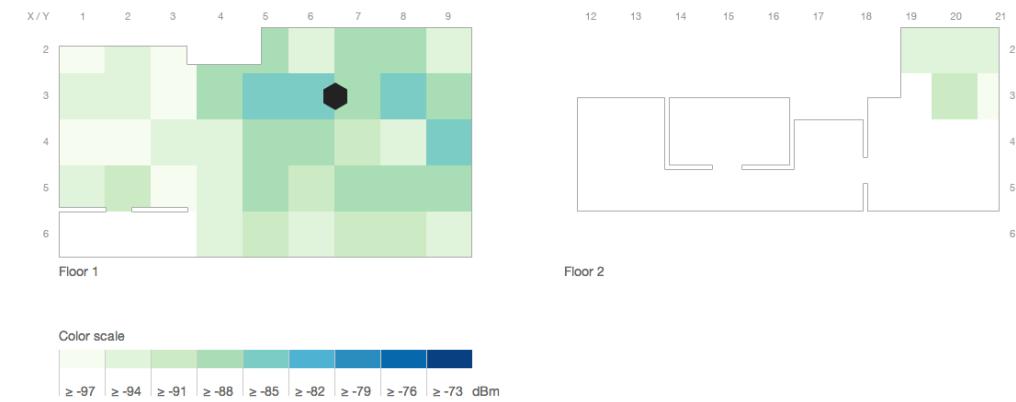


Abb. 27 Visualisierung RSSI Beacon 1, -20 dBm

Auswertung Beacon 2

Dieses Beacon befindet sich in einem Schlafzimmer. Um zu verhindern, dass das Licht im Schlafzimmer unnötig brennt wird die Signalstärke dieses Beacons auf -4 dBm festgelegt.

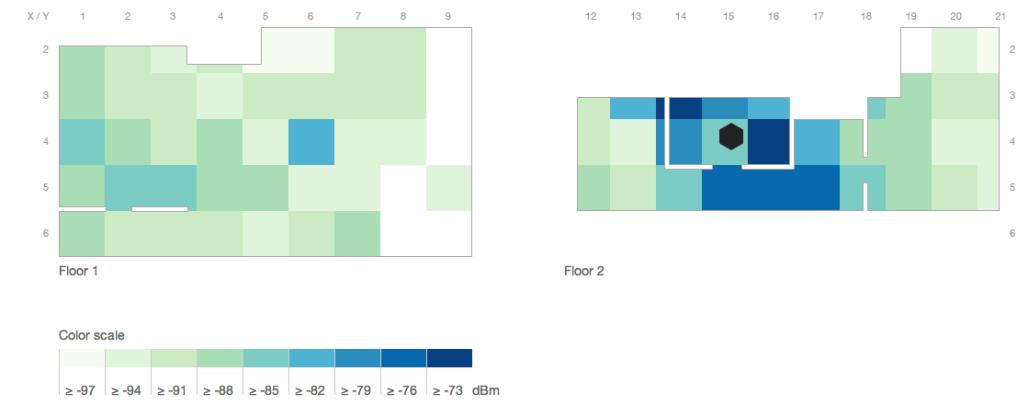


Abb. 28 Visualisierung RSSI Beacon 2, -4 dBm

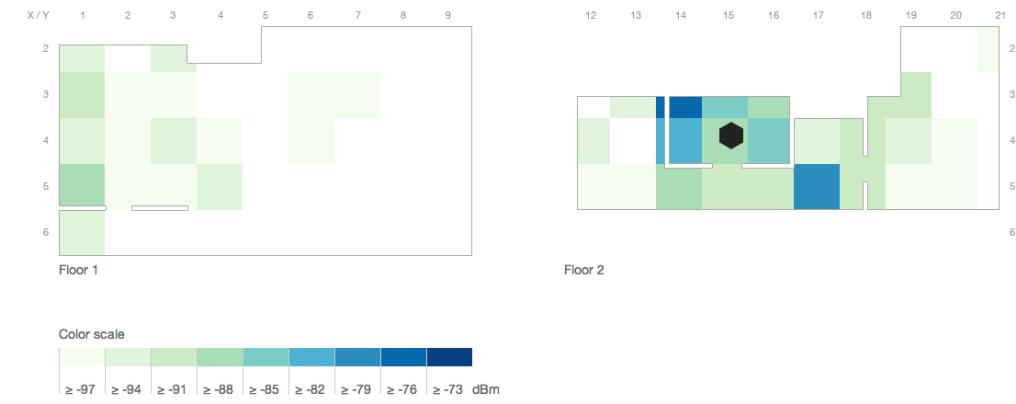


Abb. 29 Visualisierung RSSI Beacon 2, -12 dBm

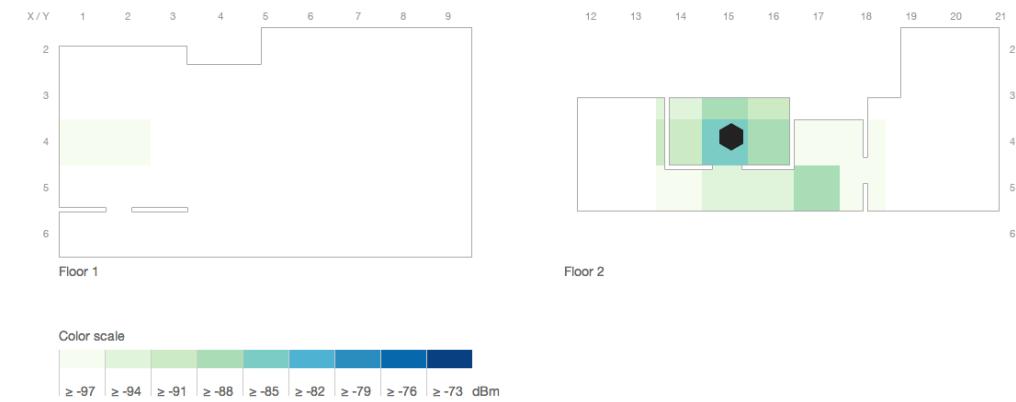


Abb. 30 Visualisierung RSSI Beacon 2, -20 dBm

Auswertung Beacon 3

Dieses Beacon befindet sich direkt im Eingangsbereich der Wohnung und soll die Beleuchtung im Gang steuern. Es wird deshalb die Signalstärke -4 dBm für dieses Beacon festgelegt.

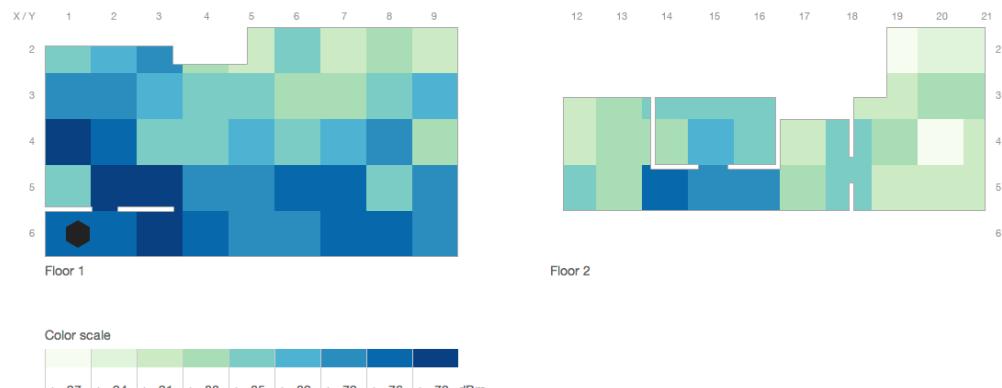


Abb. 31 Visualisierung RSSI Beacon 3, -4 dBm

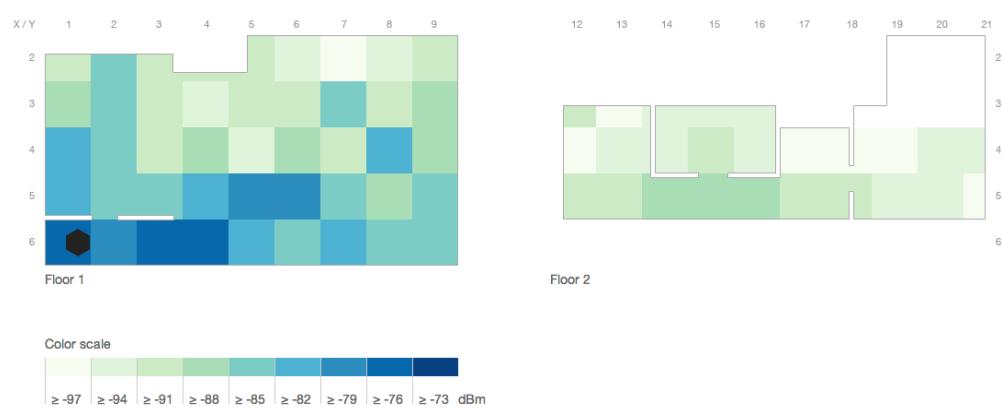


Abb. 32 Visualisierung RSSI Beacon 3, -12 dBm



Abb. 33 Visualisierung RSSI Beacon 3, -20 dBm

3 Anforderungen

3.1 Einleitung

Das manuelle Bedienen von Lichtern in einer Wohnung ist oft mühsam und ineffizient. Die Lichtschalter sind an ungünstigen Orten angebracht oder das Licht wird beim Verlassen des Hauses auszuschalten vergessen.

Es soll deshalb eine Softwarelösung entwickelt werden, welche die Lichter in einem Haus automatisch steuert. Betritt der Bewohner einen Raum, sollen die Lichter im Raum angehen - beim Verlassen des Raums sollen die Lichter wieder automatisch ausgehen. Dies spart Strom und erleichtert den Alltag.

3.2 Stakeholder

Die folgenden Stakeholder haben einen direkten oder indirekten Einfluss auf die Anforderungen. Diese Stakeholder finden sich ebenfalls im nachfolgenden Kontextdiagramm.

Stakeholder	Beschreibung
Bewohner	Der Bewohner der Wohnung ist die massgebende Zielperson. Die Anwendung soll sein Leben vereinfachen, die funktionalen Anforderungen sind deshalb auf Ihn ausgelegt.
Besucher	Besucher der Wohnung wissen im Normalfall nichts über die im Hintergrund laufende Anwendung und müssen vor Überraschungen geschützt werden.
Beacon-Hersteller	Der Beacon-Hersteller entwickelt und liefert die nötige Hardware in Form der Beacons für die Anwendung. Er hat einen Einfluss auf die technischen Anforderungen.
Lampen-Hersteller	Der Lampen-Hersteller entwickelt und produziert die steuerbaren Lampen. Er definiert die technischen Möglichkeiten der Beleuchtung.
Apple	Apple als Entwickler des iBeacon-Formats definiert die technischen Vorgaben und nimmt evtl. in Zukunft Änderungen am Format vor.

Abb. 34 Stakeholder

3.3 Systemkontext

Die Kontextabgrenzung dient zur Definition der Elemente, welche einen Einfluss auf das System haben. Das System selbst, die automatische Steuerung, wird durch den Bewohner und die Beacons beeinflusst und steuert das Licht. Besucher kommen innerhalb des Systemkontexts ebenfalls vor, haben aber keinen direkten Einfluss auf das System.

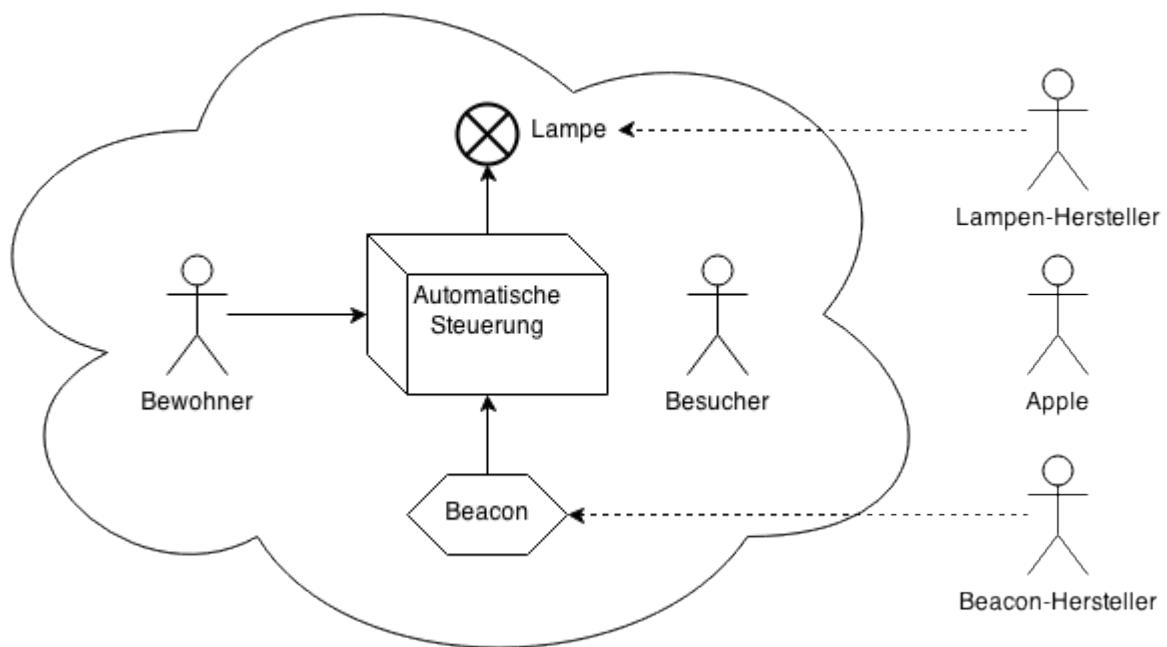


Abb. 35 Kontextdiagramm

3.4 Funktionale Anforderungen

Die funktionalen Anforderungen wurden mit der Beobachtungstechnik und anhand der eigenen Erfahrungen definiert und als Anwendungsfälle festgehalten.

Das folgender Anwendungsfalldiagramm gibt einen Überblick über die erfassten Anwendungsfälle. Es gibt zwei Akteure, den Benutzer sowie die automatische Steuerung.

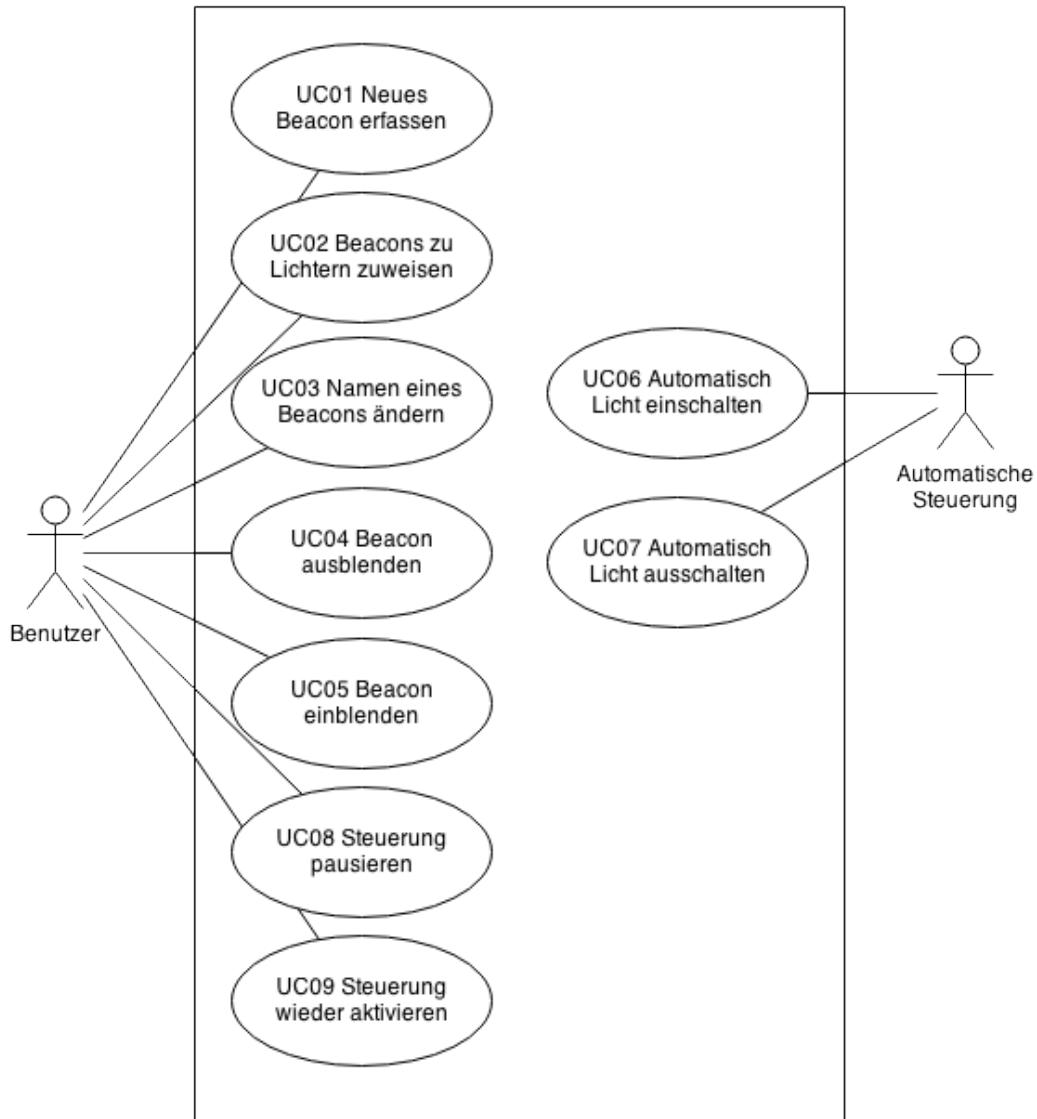


Abb. 36 Anwendungsfalldiagramm

Die Anwendungsfälle wurden mit den im folgendem Schema definierten Attributen erfasst.

Attribut	Beschreibung
Identifier	Eindeutige Nummer des Anwendungsfalls.
Name	Kurzname des Anwendungsfalls.
Priorität	Dringlichkeit des Anwendungsfalls mit dem Wertebereich tief, mittel und hoch.
Auslöser	Beschreibung des Ereignisses, welches diesen Anwendungsfall relevant macht.
Beschreibung	Kurze Zusammenfassung des Anwendungsfalls.
Vorbedingungen	Voraussetzungen welche für die Ausführung dieses Anwendungsfalls erfüllt sein müssen.
Standardablauf	Schritte zur erfolgreichen Ausführung des Anwendungsfalls.
Ergebnis	Das Resultat nach der erfolgreichen Ausführung des Standardablaufs.
Ausnahmen	Eventuell auftretende Ausnahmesituationen sowie deren Konsequenzen.

Abb. 37 Attribute Anwendungsfall

Anwendungsfälle

Identifier	UC01
Name	Neues Beacon erfassen
Priorität	mittel
Auslöser	Der Benutzer konfiguriert das System zum ersten Mal oder ist in Besitz eines neuen Beacons gekommen.
Beschreibung	Der Benutzer erfasst ein neues Beacon mit einem eigenen Namen.
Vorbedingungen	<ul style="list-style-type: none"> ▪ Der Benutzer muss die UUID sowie den Major- und den Minor-Wert des Beacons kennen. ▪ Der Benutzer ist in der Anwendung eingeloggt und befindet sich auf der Ansicht zur Konfiguration der Beacons.
Standardablauf	<ul style="list-style-type: none"> ▪ Der Benutzer klickt auf «Beacon hinzufügen». ▪ Es erscheint ein Formular zur Erfassung des Beacons. ▪ Der Benutzer füllt den Namen, die UUID, den Major- und den Minor-Wert ein. ▪ Der Benutzer klickt auf «Beacon speichern». ▪ Das Beacon wird gespeichert. ▪ Es wird wieder die Ansicht zur Konfiguration mit einer Erfolgsmeldung über die erfolgreiche Speicherung des Beacons angezeigt.
Ergebnis	Das neue Beacon erscheint auf der Konfigurationsansicht und kann einem Licht zugewiesen werden.
Ausnahmen	<ul style="list-style-type: none"> ▪ Es ist bereits ein Beacon mit der selben UUID, Major- und Minor-Werten erfasst; in diesem Fall soll das Beacon nicht gespeichert und eine Fehlermeldung angezeigt werden.

Abb. 38 UC01 Neues Beacon erfassen

Identifier	UC02
Name	Beacon zu Lichtern zuweisen
Priorität	hoch
Auslöser	Der Benutzer konfiguriert das System zum ersten Mal, ist in Besitz eines neuen Beacons oder einer neuen Lampe gekommen.
Beschreibung	Der Benutzer weisst ein Beacon zu einem oder mehreren Lichtern zu.
Vorbedingungen	<ul style="list-style-type: none"> ▪ Es wurde ein Beacon erfasst. ▪ Es ist mindestens eine Lampe vorhanden. ▪ Der Benutzer ist in der Anwendung eingeloggt und befindet sich auf der Ansicht zur Konfiguration der Beacons.
Standardablauf	<ul style="list-style-type: none"> ▪ Der Benutzer wählt die gewünschten Lichter aus. ▪ Der Benutzer klickt auf «Zuweisung speichern». ▪ Die Zuweisung der Lichter zum Beacon wird gespeichert. ▪ Es wird wieder die Ansicht zur Konfiguration mit einer Erfolgsmeldung über die erfolgreiche Speicherung der Zuweisung angezeigt.
Ergebnis	Die automatische Steuerung schaltet die ausgewählten Lichter an, sobald der Benutzer in der Nähe des Beacons ist.

Abb. 39 UC02 Beacons zu Lichtern zuweisen

Identifier	UC03
Name	Namen eines Beacons ändern
Priorität	tief
Auslöser	Ein Beacon wurde neu positioniert oder es wurde ein Schreibfehler festgestellt.
Beschreibung	Der Benutzer ändert den Namen eines bestehenden Beacons.
Vorbedingungen	<ul style="list-style-type: none"> ▪ Das Beacon wurde zuvor erfasst. ▪ Der Benutzer ist in der Anwendung eingeloggt und befindet sich auf der Ansicht zur Konfiguration der Beacons.
Standardablauf	<ul style="list-style-type: none"> ▪ Der Benutzer klickt bei dem gewünschten Beacon auf «Bearbeiten». ▪ Es erscheint ein Formular mit dem Namen des Beacons. ▪ Der Benutzer ändert den Namen des Beacons. ▪ Der Benutzer klickt auf «Beacon speichern». ▪ Das Beacon wird gespeichert. ▪ Es wird wieder die Ansicht zur Konfiguration mit einer Erfolgsmeldung über die erfolgreiche Speicherung des Beacons angezeigt.
Ergebnis	Das Beacon erscheint mit dem geänderten Namen auf der Konfigurationsansicht.

Abb. 40 UC03 Namen eines Beacons ändern

Identifier	UC04
Name	Beacon ausblenden
Priorität	tief
Auslöser	Ein Beacon wurde entfernt oder ist nicht mehr relevant.
Beschreibung	Der Benutzer markiert ein Beacon als inaktiv.
Vorbedingungen	<ul style="list-style-type: none"> ▪ Das Beacon wurde zuvor erfasst. ▪ Der Benutzer ist in der Anwendung eingeloggt und befindet sich auf der Ansicht zur Konfiguration der Beacons.
Standardablauf	<ul style="list-style-type: none"> ▪ Der Benutzer klickt bei dem gewünschten Beacon auf «Bearbeiten». ▪ Es erscheint ein Formular mit dem Status des Beacons. ▪ Der Benutzer ändert den Status des Beacons auf «Inaktiv». ▪ Der Benutzer klickt auf «Beacon speichern». ▪ Das Beacon wird gespeichert. ▪ Es wird wieder die Ansicht zur Konfiguration mit einer Erfolgsmeldung über die erfolgreiche Speicherung des Beacons angezeigt.
Ergebnis	Das Beacon wird in der Konfigurationsansicht in einer separaten Liste aufgeführt und kann nicht mehr Lichtern zugewiesen werden.

Abb. 41 UC04 Beacon ausblenden

Identifier	UC05
Name	Beacon einblenden
Priorität	tief
Auslöser	Ein Beacon wurde wieder relevant nachdem es deaktiviert wurde.
Beschreibung	Der Benutzer markiert ein Beacon als aktiv.
Vorbedingungen	<ul style="list-style-type: none"> ▪ Das Beacon wurde zuvor erfasst. ▪ Das Beacon ist inaktiv. ▪ Der Benutzer ist in der Anwendung eingeloggt und befindet sich auf der Ansicht zur Konfiguration der Beacons.
Standardablauf	<ul style="list-style-type: none"> ▪ Der Benutzer klickt bei dem gewünschten Beacon auf «Bearbeiten». ▪ Es erscheint ein Formular mit dem Status des Beacons. ▪ Der Benutzer ändert den Status des Beacons auf «Aktiv». ▪ Der Benutzer klickt auf «Beacon speichern». ▪ Das Beacon wird gespeichert. ▪ Es wird wieder die Ansicht zur Konfiguration mit einer Erfolgsmeldung über die erfolgreiche Speicherung des Beacons angezeigt.
Ergebnis	Das Beacon wird in der Konfigurationsansicht wieder normale dargestellt und kann Lichtern zugewiesen werden.

Abb. 42 UC05 Beacon einblenden

Identifier	UC06
Name	Automatisch Licht einschalten
Priorität	hoch
Auslöser	Der Benutzer kommt in die Nähe eines Beacons, welches dem Licht zugewiesen ist.
Beschreibung	Das zugewiese Licht wird eingeschaltet.
Vorbedingungen	<ul style="list-style-type: none"> ▪ Das Beacon wurde zuvor erfasst und ist aktiv. ▪ Das Beacon ist einem Licht zugewiesen. ▪ Das Licht hat Strom (wurde also nicht durch einen Lichtschalter ausgeschaltet). ▪ Der Benutzer hat die App installiert und diese erfolgreich autorisiert.
Standardablauf	<ul style="list-style-type: none"> ▪ Der Benutzer kommt in die Nähe des Beacons.
Ergebnis	Das Licht geht an und leuchtet.

Abb. 43 UC06 Automatisch Licht einschalten

Identifier	UC07
Name	Automatisch Licht ausschalten
Priorität	hoch
Auslöser	Der Benutzer entfernt sich vom Beacon, welches dem Licht zugewiesen ist.
Beschreibung	Das zugewiese Licht wird nach 3 Minuten ausgeschaltet. Diese 3 Minuten Wartezeit dienen auch zur Verhinderung von flackernden Lichtern, falls der Benutzer nur kurzzeitig (z.B. aufgrund einer Signalschwankung) ausserhalb des Bereichs gemeldet wird.
Vorbedingungen	<ul style="list-style-type: none"> ▪ Das Beacon wurde zuvor erfasst und ist aktiv. ▪ Das Beacon ist einem Licht zugewiesen. ▪ Das Licht leuchtet. ▪ Der Benutzer hat die App installiert und diese erfolgreich autorisiert.
Standardablauf	<ul style="list-style-type: none"> ▪ Der Benutzer entfernt sich vom Beacon. ▪ Der Benutzer bleibt 3 Minuten vom Beacon weg.
Ergebnis	Das Licht geht aus und leuchtet nicht mehr.
Ausnahmen	<ul style="list-style-type: none"> ▪ Der Benutzer nähert sich innerhalb von 3 Minuten wieder dem Beacon; das Licht wird in diesem Fall nicht ausgeschaltet.

Abb. 44 UC07 Automatisch Licht ausschalten

Identifier	UC08
Name	Steuerung pausieren
Priorität	mittel
Auslöser	Eine Person zusätzliche Person (z.B. ein Besucher) betritt die Wohnung und möchte nicht im Dunkeln stehen, wenn der Benutzer mit der App einen anderen Raum betritt.
Beschreibung	Der Benutzer pausiert die automatische Steuerung für eine unbegrenzte Zeit.
Vorbedingungen	<ul style="list-style-type: none"> ▪ Der Benutzer ist in der Anwendung eingeloggt und befindet sich auf der Startseite.
Standardablauf	<ul style="list-style-type: none"> ▪ Der Benutzer klickt auf «Steuerung pausieren». ▪ Der Status der Steuerung wird als pausiert angezeigt.
Ergebnis	Die automatische Steuerung der Lichter wird pausiert, es werden keine Lichter mehr automatisch an- und ausgeschaltet.

Abb. 45 UC08 Steuerung pausieren

Identifier	UC09
Name	Steuerung wieder aktivieren
Priorität	mittel
Auslöser	Die zusätzliche Person (z.B. ein Besucher) verlässt die Wohnung wieder.
Beschreibung	Der Benutzer aktiviert die automatische Steuerung wieder für eine unbegrenzte Zeit.
Vorbedingungen	<ul style="list-style-type: none"> ▪ Der Benutzer ist in der Anwendung eingeloggt und befindet sich auf der Startseite. ▪ Die Steuerung wurde pausiert.
Standardablauf	<ul style="list-style-type: none"> ▪ Der Benutzer klickt auf «Steuerung aktivieren». ▪ Der Status der Steuerung wird als aktiv angezeigt.
Ergebnis	Die automatische Steuerung der Lichter ist wieder aktiv, Lichter werden wieder automatisch an- und ausgeschaltet.

Abb. 46 UC09 Steuerung wieder aktivieren

3.5 Nichtfunktionale Anforderungen

Neben den funktionalen Anforderungen gibt es auch gewisse nichtfunktionale Anforderungen, die teilweise implizit vom Benutzer erwartet werden. Diese Qualitätsanforderungen müssen sowohl messbar wie auch

Identifier	Q01
Name	Erlernbarkeit
Beschreibung	Die Bedienung der Anwendung soll durch einen Benutzer auch ohne Anleitung innerhalb von einem Tag verstanden werden können. Vom Benutzer begangene Fehler sollen jederzeit wieder rückgängig gemacht werden können.

Abb. 47 Q01 Erlernbarkeit

Identifier	Q02
Name	Änderbarkeit
Beschreibung	Um auch zukünftige Anforderungen umsetzen zu können, soll die Anwendung erweiterbar geschrieben werden. Es muss möglich sein, weiter Funktionen einzubauen ohne bestehende zu beeinträchtigen.

Abb. 48 Q02 Änderbarkeit

Identifier	Q03
Name	Portabilität
Beschreibung	Die Anwendung soll auf einem einfachen Web-Server betrieben werden können, welcher PHP und MySQL unterstützt.

Abb. 49 Q03 Portabilität

Identifier	Q04
Name	Sicherheit
Beschreibung	Die Anwendung soll durch ein Login geschützt sein. Der Benutzer muss sich mit seinem Benutzernamen und Passwort authentifizieren, um die Anwendung zu benutzen.

Abb. 50 Q04 Sicherheit

Identifier	Q05
Name	Performance
Beschreibung	Alle Ansichten der Anwendungen sollen innerhalb von 0.9 Sekunden angezeigt werden.

Abb. 51 Q05 Performance

3.6 Benutzerschnittstellen

Die folgenden Mockups dienen dazu, die Anforderungen visuell darzustellen. Es werden die drei wichtigsten Ansichten gezeigt.

Das Dashboard wird beim Starten der Anwendung, jedesmal wenn sich der Benutzer eingeloggt hat, angezeigt. Im Dashboard erhält der Benutzer eine Übersicht über die letzten Events der Engine. Er kann zudem die Engine pausieren und wieder starten.



Abb. 52 Mockup Dashboard

Der Report zeigt dem Benutzer Informationen über die Brenndauer der Lichter während der letzten Tage an.

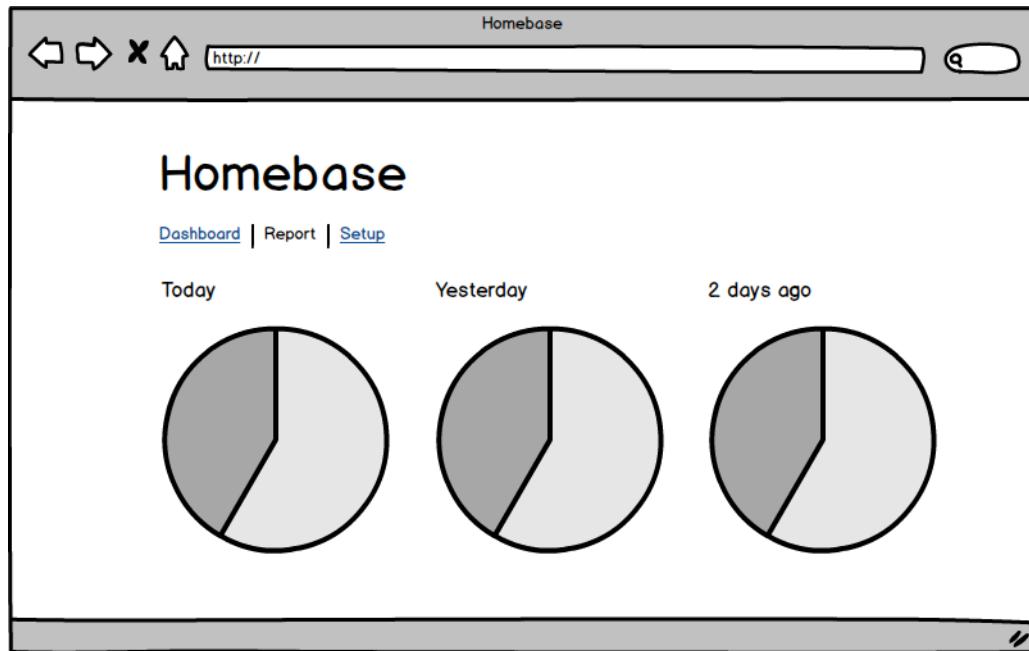


Abb. 53 Mockup Report

Im Setup kann der Benutzer die Beacons den Lichtern zuweisen.

The mockup shows a web browser window titled "Homebase". The main content area is titled "Homebase" and contains three sections for assigning lights to beacons. Each section has a diamond icon and a label: "Beacon 1 Edit", "Beacon 2 Edit", and "Beacon 3 Edit". Under each label, there is a list of three lights (Light 1, Light 2, Light 3) with checkboxes. In the first section (Beacon 1), "Light 2" is checked. In the second section (Beacon 2), "Light 2" and "Light 3" are checked. In the third section (Beacon 3), "Light 1" is checked. Below these sections is a "Save settings" button. Further down, there is a "Hidden Beacons" section with a link "Beacon 4 Edit". The browser's address bar shows "http://".

Abb. 54 Mockup Setup

4 Software

4.1 Randbedingungen

Um eine möglichst hohe Portabilität zu gewährleisten und die Anwendung auf einem einfachen Webserver betreiben zu können, soll die Anwendung mit PHP und MySQL entwickelt werden.

Das iBeacon ein von Apple entwickeltes Datenformat ist, bietet die iOS-Plattform Bibliotheken zur einfachen Umsetzung. Es soll deshalb eine iPhone-App mit Objective-C entwickelt werden.

4.2 Lösungsstrategie

Das umgesetzte Software besteht aus zwei Komponenten, einer iPhone-App sowie einer Web-Applikation. Die Aufgabe der iPhone-App ist legendlich, die Position des Benutzers zu ermitteln und diese an die Web-Applikation zu übertragen. Die Web-Applikation hat eine Anbindung an eine Datenbank und speichert dort alle Positionsangaben, steuert die Beleuchtung und stellt die gesammelten Informationen für den Benutzer dar.

Ein zentrales Element der Web-Applikation ist die *Engine*, welche die aktuellen Positionsdaten analysiert und entscheidet, welche Lichter an- und welche ausgeschaltet werden sollen.

4.3 Bausteinsicht

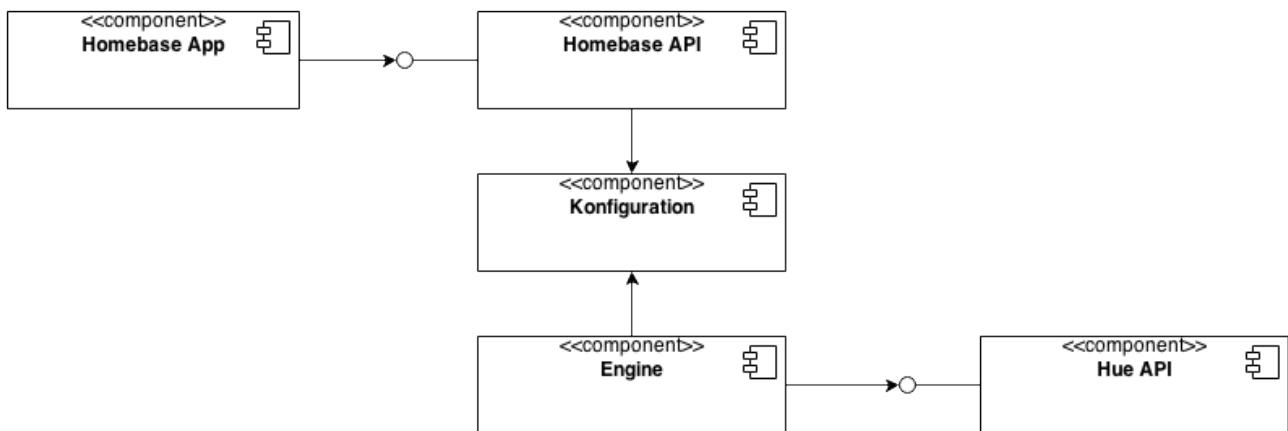


Abb. 55 Komponentendiagramm

iPhone-App

Die iPhone-App besteht nur aus einer Klasse, welche durch das Betriebssystem über das Delegate-Pattern aufgerufen wird, sobald der Benutzer den Bereich eines Beacons betritt oder verlässt. Dieses Klasse sendet dann diese Informationen über eine HTTP-Anfrage an den Server.

Web-Applikation

Die Web-Applikation wurde nach dem klassischen MVC-Pattern in PHP umgesetzt: Anfragen werden im Controller entgegengenommen, die Daten an das Model weitergeleitet und schlussendlich in der View dargestellt.

Da PHP nur wenig Funktionalität zur komplexeren Verarbeitung von HTTP-Anfragen bietet, wurde das Micro-Framework Silex [1] verwendet. Dieses stellt Funktionen für das Routing zur Verfügung und erlaubt die einfache Benutzung der Templating-Engine Twig [2].

Das nachfolgende Klassendiagramm beschreibt die Klassen im Packet `Homebase\` `Service`, welche das Model und damit die eigentliche Funktionalität enthalten.

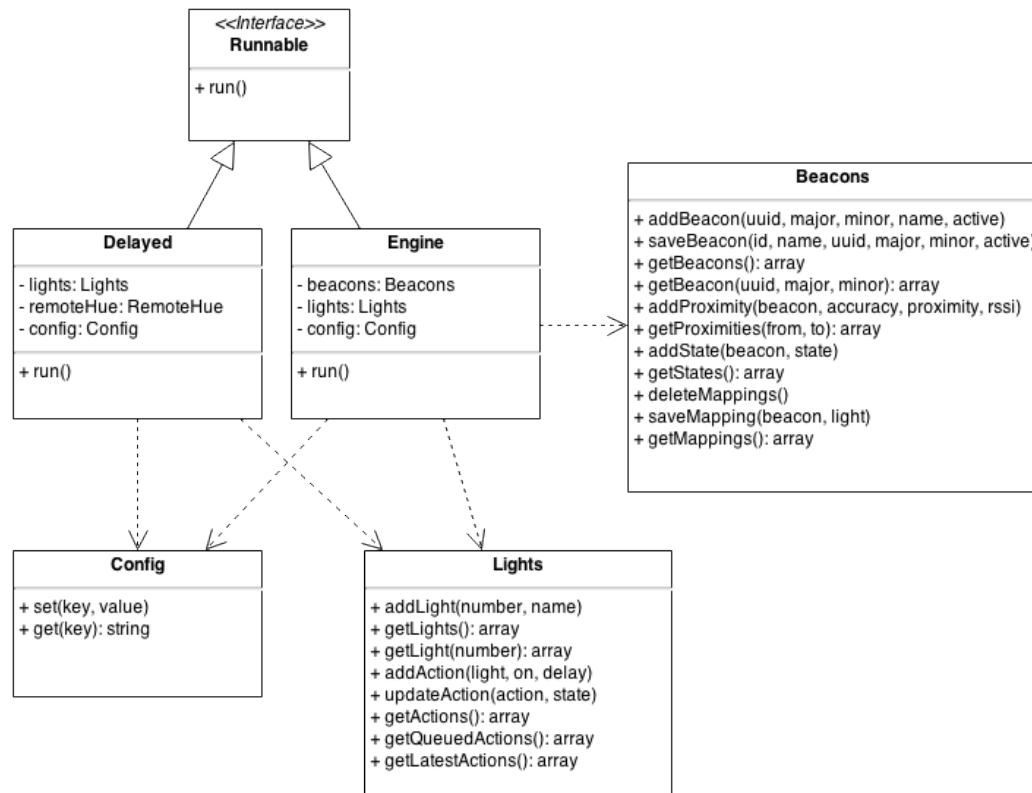


Abb. 56 Klassendiagramm `Homebase\Service`

[1] Sensio Labs (2013): Silex

[2] Sensio Labs (2014): Twig

4.4 Laufzeitsicht

Neben der Web-Oberfläche zur Konfiguration der Anwendung ist hauptsächlich die Engine, welche die automatische Steuerung der Lampen übernimmt, dynamisch. Die Funktionsweise der Engine wird in diesem Kapitel beschrieben.

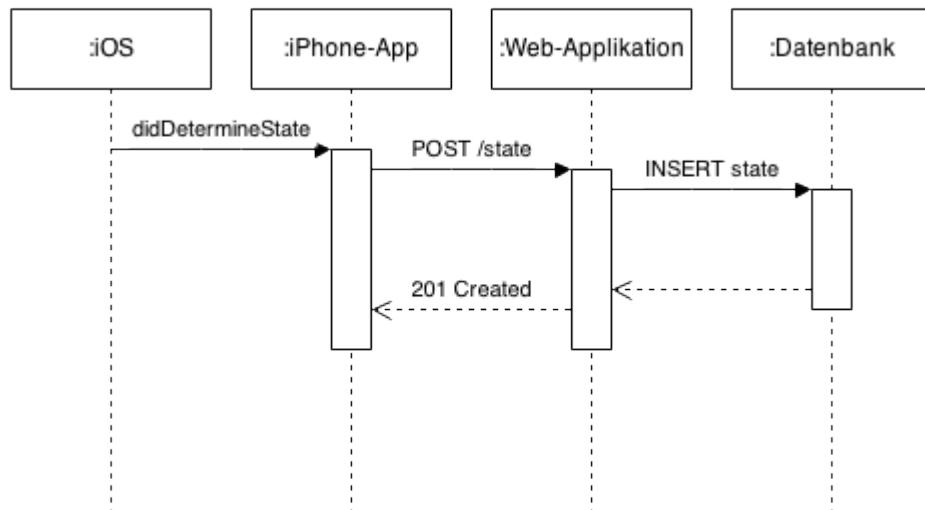


Abb. 57 Sequenzdiagramm App

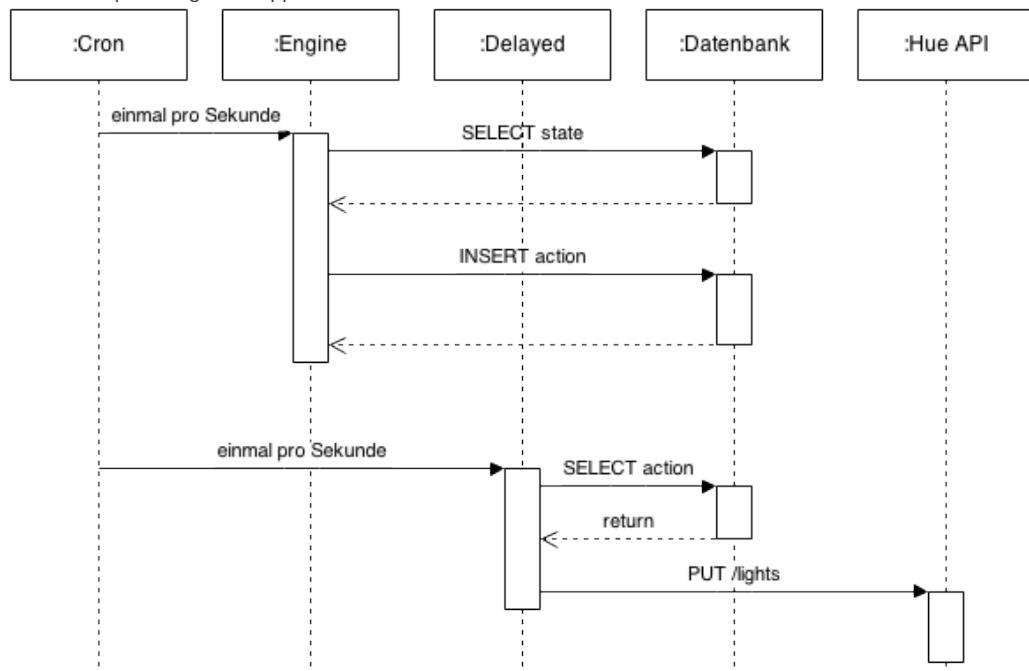


Abb. 58 Sequenzdiagramm Engine

4.5 Verteilungssicht

Die Web-Applikation wird zusammen mit dem Datenbank-Schema auf dem gleichen Server deployed. Das Deployment ist über einen Continious-Integration-Server automatisiert.

Die iPhone-App wird manuell mit einem USB-Kabel auf dem iPhone installiert.

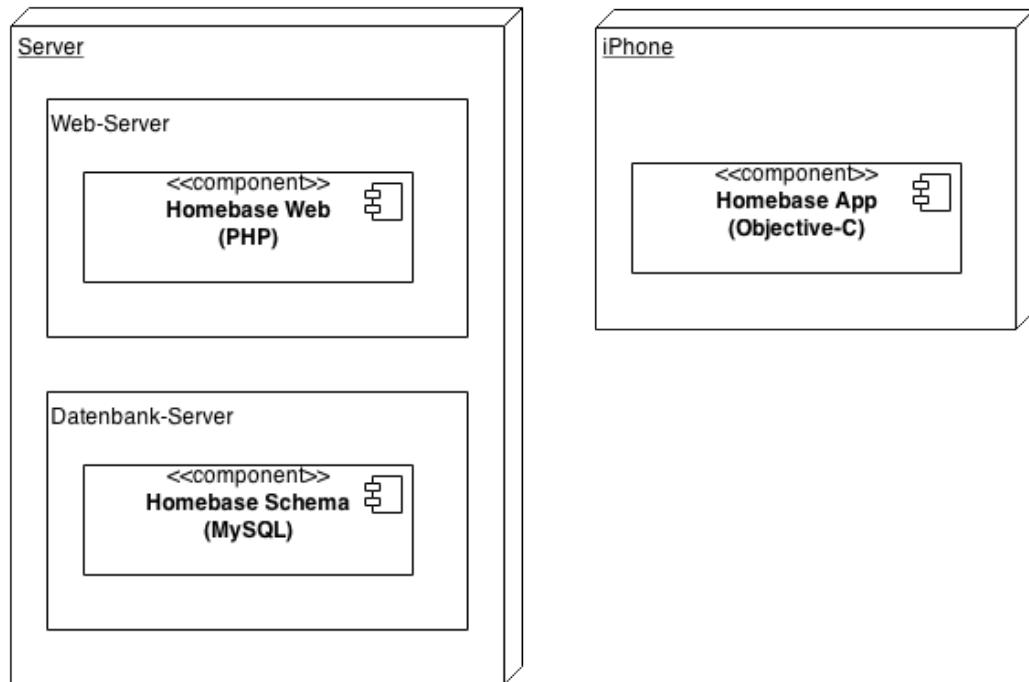


Abb. 59 Verteilungsdiagramm

4.6 Datenmodell

Sämtliche Informationen der Anwendung werden in einer MySQL-Datenbank gespeichert.

Das nachfolgende relationale Datenmodell zeigt die Entitäten sowie deren Beziehungen.

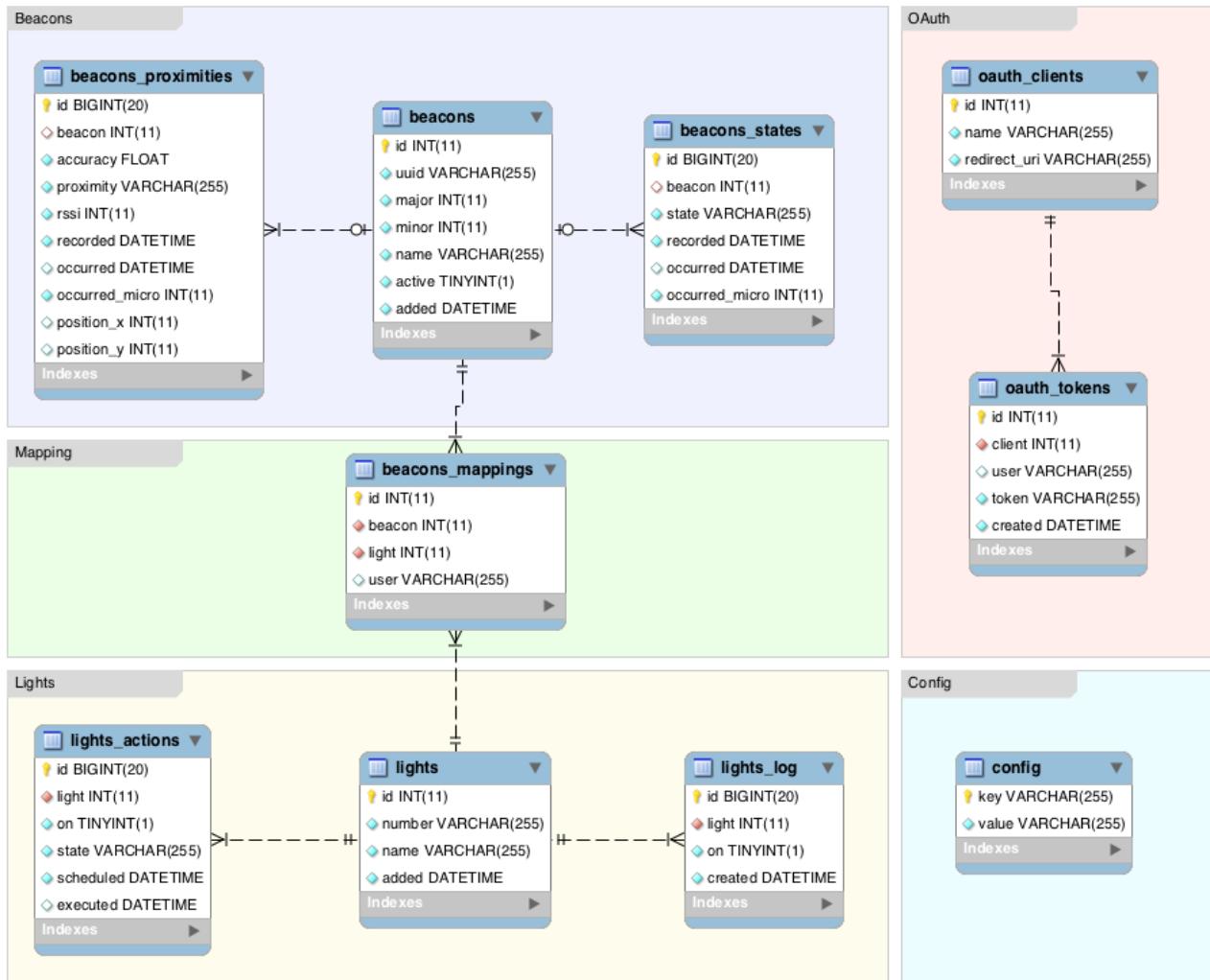


Abb. 60 Datenmodell

In der folgenden Tabelle finden sich zu jeder Tabelle eine kurze Beschreibung des Inhalts resp. des Zwecks der Tabelle.

Tabelle	Inhalt
beacons	Sämtliche vom Benutzer erfassten Beacons
beacons_proximities	Die von der iPhone-App gemessenen Abstände
beacons_states	Die von der iPhone-App gemeldeten Positionen des Benutzers
beacons_mappings	Zuweisungen der Beacons zu den Lampen
lights	Die über die hue API gefundenen Lampen
lights_actions	Durchgeführte und geplante Interaktionen mit der hue API zum An- oder Ausschalten der Lampen
lights_log	Historische Daten über die Brenndauer der Lampen
oauth_clients	OAuth 2.0 Client-Applikationen (iPhone-Apps)
oauth_tokens	Authorisierte OAuth-Tokens für die iPhone-App
config	Systemeinstellungen

Abb. 61 Tabellen Datenmodell

4.7 Benutzeroberfläche

Die Benutzeroberfläche der Web-Applikation wurde mit Bootstrap [3] umgesetzt. Diese Bibliothek ermöglicht unter anderem, dass die Web-Applikation auch auf mobilen Geräten einfach zu bedienen ist.

Das Dashboard hat neben den Events und der Schaltfläche zum pausieren der Engine auch einige statistische Informationen, wie die Anzahl an (aktiven) Beacons, die Anzahl an (leuchtenden) Lampen sowie ein Diagramm zur Nutzung der Lichter während der letzten 3 Wochen (dargestellt als Streamgraph).

[3] Mark Otto (2013): Bootstrap

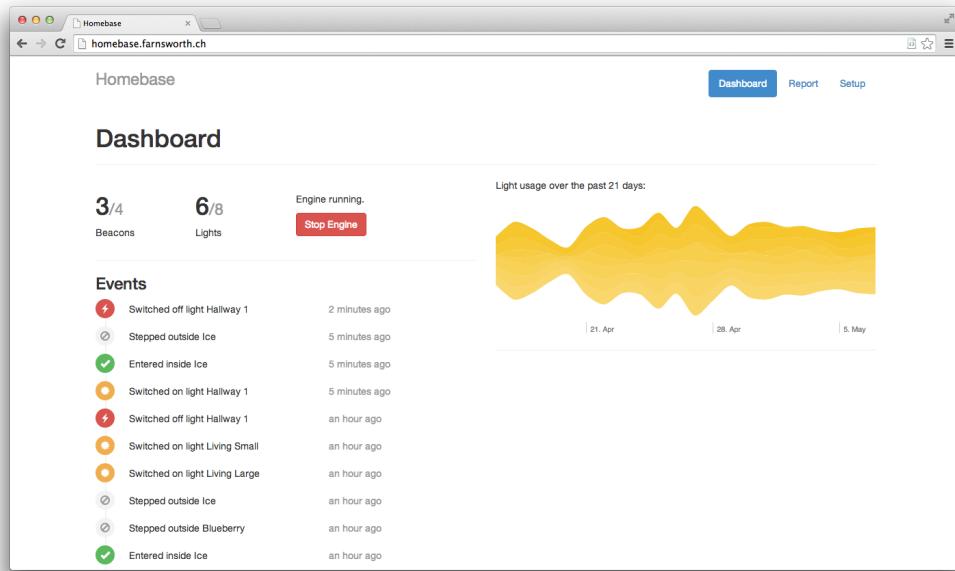


Abb. 62 Screenshot Dashboard

Der Report zeigt die Nutzung der Lichter während der letzten 3 Tage in einem Ringdiagramm. Die Darstellung wurde mit D3.js [4], einer Bibliothek zur Erstellung von Datenvisualisierungen, erstellt. Das Diagramm zeigt, zu welchen Uhrzeiten wieviele Lichter aktiv waren.

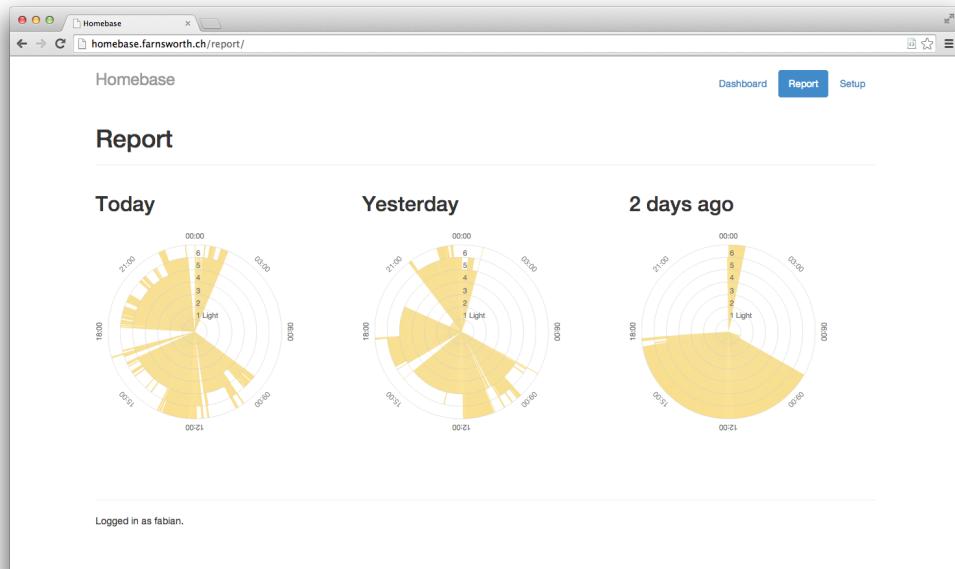


Abb. 63 Screenshot Report

[4] Mike Bostock (2013): D3.js

Auf der Seite für das Setup werden die Beacons und deren Zuweisung zu den Lichtern angezeigt und können bearbeitet werden.

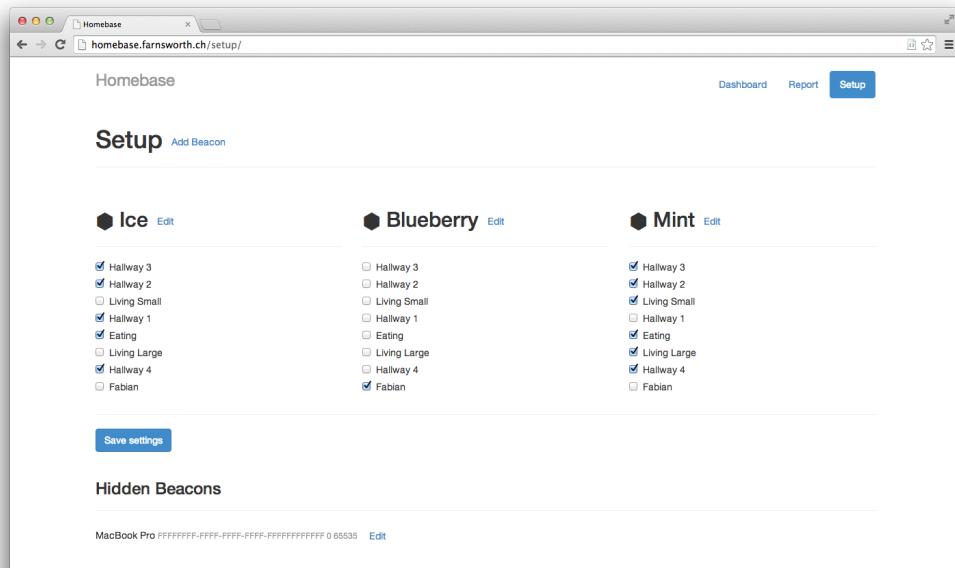


Abb. 64 Screenshot Setup

Bearbeitet der Benutzer ein Beacon, kann er dessen Namen sowie seine Identifikationsmerkmale ändern.

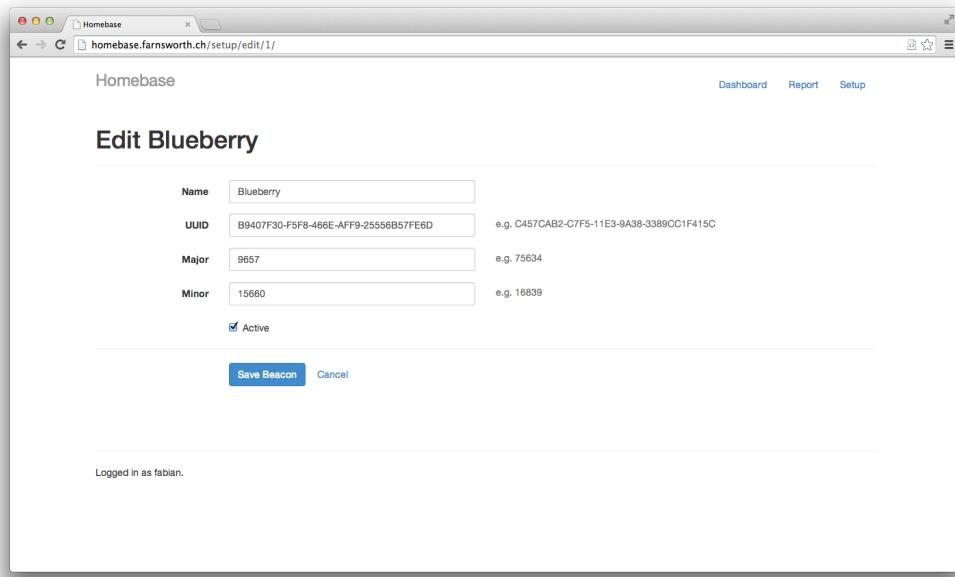


Abb. 65 Screenshot Beacon bearbeiten

5 Fazit

Die Heiamtomatisierung mit iBeacons ist durchaus möglich, muss aber detailliert geplant sein. Aufgrund der technischen Einschränkungen in iOS ist eine fortlaufende Messung der Distanz zu einem iBeacon nicht möglich, die Position des Bewohners muss deshalb über das Betreten und Verlassen des Empfangsbereichs des iPhones realisiert werden.

Die von Apple gesetzten Einschränkungen machen jedoch Sinn, da sie den Akku des iPhones schonen und einen übermässigen Energieverbrauch aufgrund einer Hintergrundaktivität verhindern. Insofern wäre es wohl auch empfehlenswert eine ähnliche Implementierung auf anderen Plattformen anzuwenden.

Aufgrund dieser Voraussetzungen ist es notwendig, die Signalstärke der Beacons auf ca. 3-5 Meter (-20 dBm) zu reduzieren. Die daraus resultierende Signalstärke ist auch aufgrund des Energieverbrauchs der Esimote Beacons empfehlenswert. Dadurch werden je nach Grösse des Raums mehrere Beacons pro Raum notwendig. Die im Projekt eingesetzte Anzahl von drei Beacons war eher knapp. Pro Raum sind drei bis vier Beacons zu empfehlen.

Für die Software gibt es bereits mehrere Ideen für eine Weiterentwicklung. Zum einen wäre es für Haushalte mit mehreren Personen sinnvoll, wenn die Anwendung auch mehrbenutzerfähig ist. In der Web-Applikation könnte die Visualisierung der stattgefundenen Events noch verbessert und die Funktionalität zur manuellen Steuerung der Beleuchtung eingebaut werden. Eine hilfreiche Funktion wäre auch die Warnung des Benutzers, falls die Batterie eines Beacons fast leer ist.

6 Anhang

6.1 Software

Die während der Arbeit verwendete Software wird hier der Vollständigkeitshalber dokumentiert.

- Mac OS X 10.9 Build 13A603
- iOS 7.1.1 Build 11D201
- Xcode 5.0.2 Build 5A3005
- Bluetooth Explorer 4.2.0 Build 4.2.0f2
- PacketLogger 4.2.0 Build 4.2.0f2
- Prince 9.0 rev 4
- MySQL Workbench 6.0 Build 6.0.7.11216
- MySQL 5.1.70
- PHP 5.4.26
- Silex 1.1.2
- PHPUnit 3.7.31
- Twig 1.15.1
- Sweet Home 3D 4.3
- Git 1.8.5.2
- Tower 1.5.4
- Bootstrap 3.1.0
- jQuery 1.9.1
- D3.js 3.4.1

6.2 Quellenverzeichnis

Apple Inc. (2013): iOS: Understanding iBeacon. 4. Dezember 2013.

<http://support.apple.com/kb/HT6048> (abgerufen am 19. April 2014)

Bluetooth SIG, Inc. (2013): Bluetooth Smart <http://www.bluetooth.com/Pages/Bluetooth-Smart.aspx> (abgerufen am 19. April 2014)

Bluetooth SIG, Inc. (2014): Company Identifiers <https://www.bluetooth.org/en-us/specification/assigned-numbers/company-identifiers> (abgerufen am 21. April 2014)

Estimote, Inc. (2013): Estimote Beacons <http://estimote.com/> (abgerufen am 4. Mai 2014)

Punch Through Design LLC (2013): LightBlue - Bluetooth Low Energy. Version 1.50. <https://itunes.apple.com/ch/app/lightblue-bluetooth-low-energy/id557428110?l=en&mt=8> (abgerufen am 20. April 2014)

Sensio Labs (2013): Silex <http://silex.sensiolabs.org/> (abgerufen am 4. Mai 2014)

Sensio Labs (2014): Twig <http://twig.sensiolabs.org/> (abgerufen am 4. Mai 2014)

Mark Otto (2013): Bootstrap <http://getbootstrap.com/> (abgerufen am 7. Mai 2014)

Mike Bostock (2013): D3.js <http://d3js.org/> (abgerufen am 7. Mai 2014)

eTeks (2014): Sweet Home 3D <http://www.sweethome3d.com/> (abgerufen am 7. Mai 2014)

Apple Inc. (2013): Hardware IO Tools for Xcode. Veröffentlicht am 22. Oktober 2013. <https://developer.apple.com/downloads/index.action?name=hardware%20io> (abgerufen am 20. April 2014)

Bluetooth SIG, Inc. (2010): Bluetooth Specification Version 4.0. https://www.bluetooth.org/docman/handlers/downloaddoc.ashx?doc_id=229737 (abgerufen am 22. November 2013)

Bluetooth SIG, Inc. (2013): Assigned Numbers <https://www.bluetooth.org/en-us/specification/assigned-numbers> (abgerufen am 22. November 2013)

Bluetooth SIG, Inc. (2012): Supplement to Bluetooth Core Specification. Version 2. https://www.bluetooth.org/docman/handlers/downloaddoc.ashx?doc_id=260933 (abgerufen am 22. November 2013)

Apple Inc. (2011): CoreBluetooth: Health Thermometer. Version 1.0.

https://developer.apple.com/library/mac/samplecode/HealthThermometer/Introduction/Intro.html#/apple_ref/doc/uid/DTS40011370 (abgerufen am 20. November 2013)

Apple Inc. (2013): Location and Maps Programming Guide https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/LocationAwarenessPG/Introduction/Introduction.html#/apple_ref/doc/uid/TP40009497 (abgerufen am 24. November 2013)

Apple Inc. (2013): Core Bluetooth Programming Guide. 18. September 2013.

https://developer.apple.com/library/ios/documentation/NetworkingInternetWeb/Conceptual/CoreBluetooth_concepts/CoreBluetooth_concepts.pdf (abgerufen am 20. November 2013)

Laurent Gaches (2013): BeaconEmitter. <https://github.com/lgaches/BeaconEmitter> (abgerufen am 20. November 2013)

6.3 Abbildungsverzeichnis

Abb. 1 Projektablauf	6
Abb. 2 Soll / Ist Vergleich Aufwand	6
Abb. 3 iBeacon Logo	7
Abb. 4 GATT Profilhierarchy	8
Abb. 5 Aufbau Core Bluetooth	8
Abb. 6 Rolen in Bluetooth Low Energy	9
Abb. 7 Beispielanwendung von Core Bluetooth	9
Abb. 8 Screenshot LightBlue	11
Abb. 9 Beispielanwendung von Core Bluetooth	11
Abb. 10 Screenshot LightBlue iBeacon	12
Abb. 11 Screenshot Bluetooth Explorer	13
Abb. 12 Screenshot PacketLogger	14
Abb. 13 Advertisement Daten iBeacon	14
Abb. 14 Struktur Advertising and Scan Response	15
Abb. 15 Bluetooth Advertisement iBeacon	16
Abb. 16 Screenshot Bluetooth Explorer nach Scan	17
Abb. 17 Foto Estimote Beacon	18
Abb. 18 Visualisierung erster Stock	19
Abb. 19 Visualisierung zweiter Stock	20
Abb. 20 Screenshot iPhone-App	21
Abb. 21 Fotos Messung	21
Abb. 22 Protokoll Messung 1	23
Abb. 23 Protokoll Messung 2	23
Abb. 24 Protokoll Messung 3	23
Abb. 25 Visualisierung RSSI Beacon 1, -4 dBm	24
Abb. 26 Visualisierung RSSI Beacon 1, -12 dBm	24
Abb. 27 Visualisierung RSSI Beacon 1, -20 dBm	24
Abb. 28 Visualisierung RSSI Beacon 2, -4 dBm	25
Abb. 29 Visualisierung RSSI Beacon 2, -12 dBm	25
Abb. 30 Visualisierung RSSI Beacon 2, -20 dBm	25
Abb. 31 Visualisierung RSSI Beacon 3, -4 dBm	26
Abb. 32 Visualisierung RSSI Beacon 3, -12 dBm	26
Abb. 33 Visualisierung RSSI Beacon 3, -20 dBm	26
Abb. 34 Stakeholder	27
Abb. 35 Kontextdiagramm	28
Abb. 36 Anwendungsfalldiagramm	29

Abb. 37 Attribute Anwendungsfall	30
Abb. 38 UC01 Neues Beacon erfassen	31
Abb. 39 UC02 Beacons zu Lichtern zuweisen	32
Abb. 40 UC03 Namen eines Beacons ändern	32
Abb. 41 UC04 Beacon ausblenden	33
Abb. 42 UC05 Beacon einblenden	33
Abb. 43 UC06 Automatisch Licht einschalten	34
Abb. 44 UC07 Automatisch Licht ausschalten	34
Abb. 45 UC08 Steuerung pausieren	35
Abb. 46 UC09 Steuerung wieder aktivieren	35
Abb. 47 Q01 Erlernbarkeit	36
Abb. 48 Q02 Änderbarkeit	36
Abb. 49 Q03 Portabilität	36
Abb. 50 Q04 Sicherheit	36
Abb. 51 Q05 Performance	37
Abb. 52 Mockup Dashboard	38
Abb. 53 Mockup Report	39
Abb. 54 Mockup Setup	39
Abb. 55 Komponentendiagramm	40
Abb. 56 Klassendiagramm Homebase\Service	41
Abb. 57 Sequenzdiagramm App	42
Abb. 58 Sequenzdiagramm Engine	42
Abb. 59 Verteilungsdiagramm	43
Abb. 60 Datenmodell	44
Abb. 61 Tabellen Datenmodell	45
Abb. 62 Screenshot Dashboard	46
Abb. 63 Screenshot Report	46
Abb. 64 Screenshot Setup	47
Abb. 65 Screenshot Beacon bearbeiten	47

6.4 Glossar

Beacon: Ein elektronisches Bauteil, das ein iBeacon-Signal aussendet.

RSSI: Received Signal Strength Indicator, gibt die empfange Stärke eines Signals an.

Continious-Integration: Automatisiertes Testen von Software in regelmässigen Abständen.

Deployment: Das Verschieben oder Aktualisieren von Software auf die Ziellaufzeitumgebung.

Template-Engine: Interpretiert einen einfachen Syntax im HTML-Code und verbindet diesen mit Daten zu einer fertigen HTML-Seite.

API: Application Programming Interface, bezeichnet die Schnittstelle, über welche der Client Daten vom Server abruft.

Client: Ein Anwendungsprogramm, welches auf dem Gerät des Benutzers läuft und meist mit einem Server kommuniziert.

REST: Representational State Transfer, ist eine Architektur für eine API, welche die nach HTTP definierten Regeln befolgt.

Bundle: Zusammenfassung mehrerer Klassen und Konfigurationsdateien innerhalb eines Symfony-PHP-Projekts.

HTTP: Hypertext Transfer Protocol, ist ein Protokoll zur Übertragung von Daten über ein Netzwerk.

HTTP-Header: Feld einer HTTP-Übertragung, welches zusammen mit den Daten im HTTP-Protokoll übertragen wird.

JSON: JavaScript Object Notation, ist ein kompaktes, lesbaren Datenformat, welches oft im Internet verwendet wird.

XML: Extensible Markup Language, ist ebenfalls ein Datenformat, welches oft im Internet Verwendung findet, jedoch mächtiger ist als JSON.

Desktop: Computer, welche von einem Benutzer bedient meist an einem Schreibtisch bedient wird.

Browser: Software zur Darstellung von Websites aus dem Internet. Beispielsweise Google Chrome oder Mozilla Firefox.

URL: Uniform Resource Locator, eine Internet-Adresse welche auf eine Ressource wie z.B. eine Website zeigt.

Code Coverage: Beschreibt, wie viel Prozent des Source-Codes durch Unit Tests abgedeckt werden.

iOS: Betriebssystem von Apple für Handys und Tablets (iPhone und iPad).

Micro-Framework: Bezeichnet eine Mischung ein Framework mit reduzierter Funktionalität, welche eher wie eine Bibliothek aufgebaut ist.