DAT-Projekt Lichtsteuerung

Marius **Schuller**
Stefan **Thiemann**
Patrick **Wildt**

11. November 2015

Inhaltsverzeichnis

Hier wird kurz die Grundidee des Schwerpunktprojekts, sowie welche Hardware benötigt wird, beschrieben.

1 Einführung

Im Zuge der Internet-of-Things-Kampagne[1] werden immer mehr "dumme" bzw. einfache Geräte miteinander vernetzt und die resultierenden Daten intelligent miteinander verknüpft. Dazu gehören auch Lichter und Glühbirnen. Zur Vernetzung und Steuerung der Lichter existieren bereits mehrere aktuelle Technologien. Mit Hilfe einer der standardisierten Technologie möchten wir einen Controller implementieren, welcher diese Lichter kontrollieren kann.

2 Lichtsteuerungs-Technologien

Üblicherweise möchten Hersteller ein eigenes Produkt-Ökosystem erstellen, aus dem ein Anwender nicht oder nur schwer entkommen kann. Hierfür werden von den Herstellern eigene, unfreie Protokolle implementiert. Beispielsweise bietet *LimitlessLED* [2] Glühbirnen, welche sich über 2,4 GHz WLAN in das lokale Netzwerk verbinden. Für die eigentliche Steuerung wurde dazu eine eigene API entwickelt. Eine weitere bekannte Technologie zur Steuerung von Geräten ist *Bluetooth*. Hier ist es derzeit möglich mit Hilfe des *Generic Attribute Profile*, kurz *GATT*[3], ein eigenes Protokoll zu sprechen. Dies wird bei mehreren smarten Glühbirnen verwendet um ein proprietäres Lichtsteuerungsprotokoll zu implementieren.

Die *Bluetooth* Konkurrenten *Z-Wave*[4], welches sich auf das so genannte *Home Control*-Szenario konzentriert, sowie *ZigBee*[5], implementieren wieder jeweils eigene Lichtprotokolle. Diese Protokolle sind jedoch für jeden Client des Funkstandards nutzbar, sodass die Hersteller kein eigenes Protokoll implementieren mussten. Der Funkstandard *ZigBee* wird von den namhaften Herstellern *Philips* und *Osram* verwendet.

Für das Schwerpunktprojekt würden wir uns auf *ZigBee* kompatible Geräte konzentrieren. Vor allem die Produkte der *Philips hue* Reihe.

[1] `http://www.nextgenerationmedia.de`
[2] `http://www.limitlessled.com`
[3] `https://de.wikipedia.org/wiki/Bluetooth-Profile`
[4] `http://www.z-wavealliance.org`
[5] `http://www.zigbee.org`

3 Komponenten

Die eigentliche Logik zur Steuerung der Lichter kann auf einem *RaspberryPi* implementiert werden. Um den Funkstandard *ZigBee* sprechen zu können wird ein kompatibles Funkmodul benötigt. Hierfür kann das *RaspBee*-Modul verwendet werden. Dieses gibt es in zwei Varianten, *Basic* und *Premium*. Während man mit der Basic-Variante nur mit 5 Knoten sprechen darf, ist dies bei der Premium-Variante unbegrenzt. Die Lichter würden aus einem *Philips Hue* Starterkit bestehen.

4 Hardware

4.1 RaspBee Premium, Raspberry-Pi Einzeln

Menge	Produkt	Einzelpreis	Gesamtpreis
3	RaspberryPi 2	42 Euro	126 Euro
3	RaspBee Premium	60 Euro	180 Euro
3	Philips Hue LED 1 x 9W A60 E27	59 Euro	177 Euro
Gesamtpreis			**483 Euro**

4.2 RaspBee Premium, Raspberry-Pi Bundle

Menge	Produkt	Einzelpreis	Gesamtpreis
3	RaspberryPi 2 Bundle	70 Euro	210 Euro
3	RaspBee Premium	60 Euro	180 Euro
3	Philips Hue LED 1 x 9W A60 E27	59 Euro	177 Euro
Gesamtpreis			**567 Euro**

4.3 Hinweis

Unter Umständen sind Bestandteile der Liste schon im Vorrat der Hochschule oder der Projektteilnehmer. Je nach Beteiligung der Fachhochschule würden wir für einen Teil der Kosten aufkommen.

5 Grobarchitektur

In Abbildung 1 wird der grobe Ablauf der Kommunikation unter allen involvierten Komponenten beschrieben.

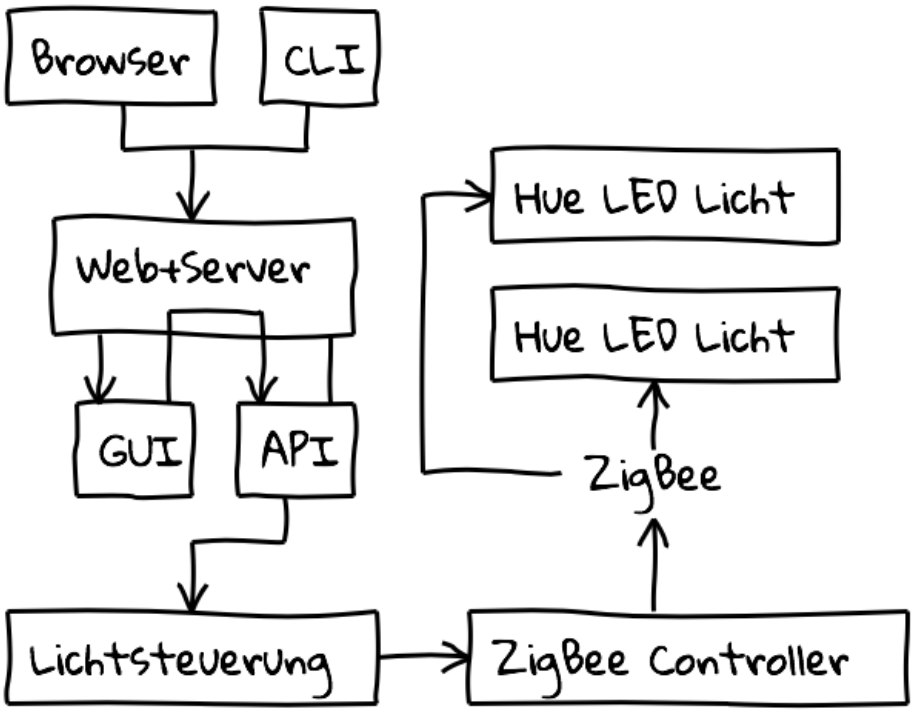

Abbildung 1: Kommunikationsablauf der Komponenten

Die Lichtsteuerung soll aus mehreren miteinander interagierenden Komponenten bestehen, welche im Weiteren genauer beschrieben werden.

5.1 Hue LED Licht

Die Lampen, welche gesteuert werden sollen, werden in eine herkömmliche Fassung geschraubt worüber sie wie normale Leuchtmittel mit Strom versorgt werden. Die Hue LEDs besitzen außerdem einen ZigBee-Chip, mit dem sie Teil eines ZigBee-Netzwerks werden können. In diesem Netzwerk arbeiten sie als *End Device*. Über das ZigBee *Light Link*-Protokoll können die Lampen angesprochen werden und bestimmte Einstellungen wie Helligkeit und Farbe eingestellt werden.

5.2 ZigBee Controller

Der ZigBee Controller ist die eigentliche Funkeinheit. Sie stellt eine rohe Programmierschnittstelle bereit, um auf das ZigBee-Netzwerk zugreifen zu können.

Abbildung 2: RaspBee-Funkmodul

Der Controller besteht aus zwei Komponenten. Zum einen dem *RaspBee*, eine aufsteckbare Erweiterungsplatine mit Funkmodul für *Raspberry Pi*, und zum anderen dem *Raspberry Pi* selber. Der *Raspberry Pi*, ein Entwicklungsboard, besitzt eine Reihe an GPIO Pins am Rand des Boards.

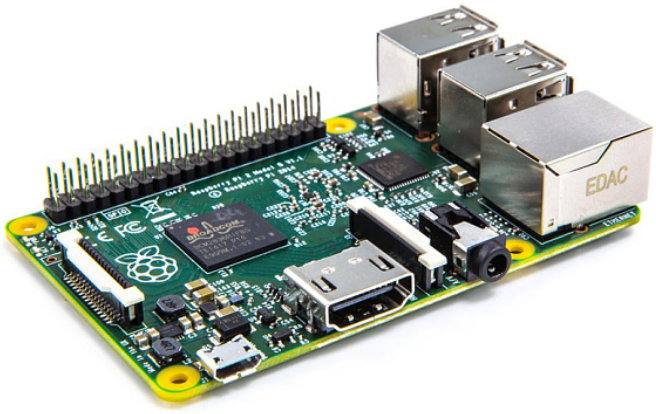

Abbildung 3: Raspberry 2

Das *RaspBee* ist an diese GPIO Pins angepasst und wird dadurch mit Strom gespeist. Weiterhin werden die *UART*-Pins zur seriellen Kommunikation mit einem Treiber, der auf dem *Raspberry Pi* betrieben wird, verwendet.

5.3 Lichtsteuerung

Die Lichtsteuerung dient als hardwarenahes Backend. Diese Software, welche auf dem *Raspberry Pi* betrieben wird, kommuniziert über die serielle Schnittstelle mit dem ZigBee Controller. Um auf die Nodes des Netzwerks zugreifen zu können bietet das Backend eine Programmierschnittstelle an.

5.4 API

Die API ist eine weitere Software auf dem *Raspberry Pi*. Sie verwendet die Schnittstelle der Lichtsteuerung und bietet eine REST-basierte Webschnittstelle an. Diese Schnittstelle bietet einen vereinfachten Zugriff auf das Funknetz, mit Fokus auf Steuerung und Verwaltung der Lampen. Durch die Trennung der einzelnen Programme bleiben Treiber, API und GUI austauschbar.

5.5 Webserver

Der Webserver dient primär zum Zugriff auf die Web-GUI, welche die REST-API über den Browser zugänglich macht um bequem die Lichter verwalten und steuern zu können. Weiter wird hier die API etwaigen CLI-Programmen zur Verfügung gestellt.

5.6 GUI

Die GUI, im HTML5 Standard, ist die Schnittstelle zum Benutzer und wird im Browser dargestellt. Über diese kann der Benutzer die Nodes steuern, wie z.B. Helligkeit und Farbe einstellen. Zum bequemen Verwenden der API aus dem Browser wird JavaScript auf dem Client verwendet.

5.7 CLI

Um die Lampensteuerung automatisieren zu können, etwa um aus dem Urlaub oder zeitgesteuert in seinem Haus das Licht aus- oder anzuschalten, ist ein Kommandozeileninterface wünschenswert. Für viele Sprachen gibt es bereits Module mit denen REST-APIs angesprochen werden können, daher ist die Programmiersprache in der das CLI-Programm umgesetzt werden soll zweitrangig.

6 Umsetzung

Umgesetzt und von uns implementiert werden soll die API sowie die CLI. Eine rudimentäre Umsetzung der Web-GUI ist auch denkbar um die prinzipielle Möglichkeit zu verdeutlichen.

Die Treibersoftware um das *RaspBee*-Modul anzusteuern ist leider nicht Open Source. Diese Software bietet eine Programmierschnittstelle, um welche wir unsere API herum entwickeln können. Der *RaspBee*-Hersteller hat auf Nachfrage bestätigt, dass weder die UART-Spezifikation, noch der Quellcode des Treibers öffentlich sei. Da ein Reverse-Engineering des Treibers den Rahmen des Projekts sprengen würde haben wir uns darauf geeinigt, die Software des Herstellers zu verwenden.

6.1 Einschränkungen

Nach anfänglicher Exploration der mit der *RaspBee*-Modul zugehörigen Software von *Dresden Elektronik* wurde festgestellt, dass viel der von uns zu implementierenden Funktionalität bereits in der bereitgestellten *deCONZ*-Software vorhanden war. Die Anfrage an den Support von *Dresden Elektronik* nach Zugang zur Dokumentation der *UART*-Schnittstelle und dem Funkmodul wurde verweigert.

Weiter war das Vorhaben aus Punkt 5 inhaltlich zu groß gefasst als dass es in der gegebenen Zeit vollumfänglich durchgeführt werden könnte.

7 Installation

7.1 Betriebssystem

Das im raspberry Pi Bundle vorinstallierte System ist NOOBS. Dieses System dient als Platform um das eigentliche System zu installieren. Es stellt eine grafische Oberfläche auf der HDMI-Schnittstelle bereit auf der das zu installierende Betriebssystem ausgewählt werden kann.

Da die deCONZ-Software nur in Binärform vorliegt muss ein daran angepasstes Betriebssystem installiert werden. Das Raspbian[6], eine für das Raspberry Pi angepasste Debian Version, ist in der Version Wheezy zur deCONZ-Software kompatibel. Weiterhin ist Raspbian eines der verbreitetsten Systeme für den raspberry Pi.

Alternativ kann Raspbian manuell installiert werden. Dafür muss ein

[6] `https://www.raspbian.org/`

Image[7] der Version Wheezy heruntergeladen werden. Das Image ist als Zip komprimiert und muss nach dem Entpacken auf die SD-Karte gespielt werden. Das kann mit folgenden Beispielkommandos ausgeführt werden.

Listing 1: Raspbian manuell installieren

```
unzip 2015-05-05-raspbian-wheezy.zip
dd if=2015-05-05-raspbian-wheezy.img of=/dev/disk
```

Der raspberry Pi bekommt automatisch von einem im Netzwerk verbundenen DHCP- Server eine IP-Adresse. Diese kann nun entweder über die grafische Oberfläche abgefragt oder aus dem DHCP-Server ausgelesen werden. Mit Hilfe dieser Information ist es nun möglich sich auf den automatisch gestarteten SSH-Server einzuwählen.

7.2 deCONZ

[7] https://www.raspberrypi.org/downloads/raspbian/