HTW-BERLIN

Thema:

Daten von Sensoren ermitteln und mittels ESP8266 Microkontroller an Raspberry Pi senden und auswerten.

Fachbereich : Angewandte Informatik

Themensteller: Prof. Dr. Alexander Huhn

Vorgelegt von: s0546864 - Gires Ntchouayang Nzeunga  
s0532766 - Nikiema Wendpanga Mohamed  
s0546864 - Sylwester Korpik

Abgabetermin: 05.01.2018

**Inhaltsverzeichnis**

**1 Einleitung**

**2 Grundlagen**

**3 Umsetzung**

**4 Ausblick**

**5 Literatur**

**6 Anhang**

**1. Einleitung**

1999 wurde der Begriff „Internet of Things“ das erste Mal von Kevin Ashton verwendet und bezeichnet eine Infrastruktur die es ermöglich Gegenstände miteinander zu vernetzen und zusammenarbeiten zu lassen. Seitdem wäscht das Internet der Dinge stetig und ermöglicht immer mehr Geräten sich miteinander zu vernetzen und Daten auszutauschen. Um diese Daten auszutauschen braucht man ein geeignetes Protocol zur Übertragung wie zum Beispiel HTTP. Das Problem besteht darin das HTTP für diesen gebrauch nicht geeignet ist da es unter anderem zu große Datenmengen verschickt und zu langsam ist.

Die Lösung für dieses Problem lautet MQTT. MQTT ist ein offenes Nachrichtenprotokoll für „Machine-to-Machine“ Kommunikation. Unser Projekt beschäftigt sich mit diesem Protokoll. Die Aufgabe besteht darin einen MQTT Broker einzurichten der als Server fungiert. Der MQTT Broker soll die Daten vom ESP8266 erhalten und eine Visualisierung dieser Daten ermöglichen. Die Daten sollen zwei ESP-Module von Sensoren erhalten, unter anderem von einem Luft- und Feuchtigkeitssensor sowie einem Ultraschallsensor. Diese sollen die Daten über einen Hotspot weiter an den Broker leiten, der die Daten auswertet. Die ESP Module arbeiten hier als Client im MQTT Protocol.

**2. Grundlagen**

**2.1. IoT Grundlagen**

Das Konzept des Internets der Dinge ist seit mehr als einem Jahrzehnt Gegenstand von Forschung, aber dennoch sind viele Aspekte nicht klar definiert. Zum Beispiel gibt es heute keine standardisierte und spezifische Architektur für IoT.

Trotz dieser mangelnden Kompatibilität gibt es eine allgemein bekannte Dreischicht-Architektur (Abbildung 1). Diese Schichten sind: die Wahrnehmungsschicht, die Netzwerkschicht und die Anwendungsschicht.

Abbildung 1: Die drei Schichten eines IOT-Modells

**2.1.1. Die Ebene der Wahrnehmung**

Die Hauptaufgabe der Wahrnehmungsschicht besteht darin, physikalische Eigenschaften wie Temperatur, Feuchtigkeit, Lichtstärke, Geschwindigkeit usw. durch verschiedene Detektionsvorrichtungen zu erkennen und diese Information in digitale Signale umzuwandeln. Die Objekte dieser Schicht können Erkennungsfähigkeiten und / oder Betätigungsfähigkeiten aufweisen. (Ein Aktor ist ein Gerät, das programmierte Befehle empfangen und Aufgaben zu bestimmten Zeiten ausführen kann).

**2.1.2. Die Netzwerkschicht**

Die Netzwerkschicht ist die Schicht, die für das Übertragen von Daten, die von der Wahrnehmungsebene empfangen werden, zu einer Datenbank, einem Server oder einem Verarbeitungszentrum verantwortlich ist. Die wichtigsten Technologien, um diese Ebene zu erreichen, sind: 2G / 3G / LTE-Mobilfunktechnologien, Wi-Fi, Bluetooth, Zigbee oder Ethernet. Mit diesen verschiedenen Technologien können wir mehrere Objekte bearbeiten, die in Zukunft verbunden werden .

Das Internet der Dinge wird ein riesiges Netzwerk sein, das nicht nur eine Vielzahl von Objekten verbindet, sondern auch heterogene Netzwerke umfasst.

**2.1.3. Die Anwendungsschicht**

Die Anwendungsschicht analysiert die von der Netzwerkschicht empfangenen Informationen. Diese Ebene bietet Anwendungen für alle Arten von technologischen Herausforderungen. Diese Anwendungen favorisieren das Internet der Dinge, weshalb diese Schicht eine wichtige Rolle bei der Verbreitung von IoT spielt.

**2.2.** **Die Komponenten eines IOT-Modells**

Alle für den Aufbau eines IoT- Modells notwendigen Komponenten werden im Folgenden anhand des Dreischichtmodells klassifiziert (siehe Abbildung 2):

Die Wahrnehmungsebene besteht aus:

* Sensoren: Erkennen physikalischer Eigenschaften und Konvertieren dieser Eigenschaften in digitale Signale.
* Die Aktuatoren: Empfangen von Befehlen zum Ausführen von Aktionen an bestimmte Momente.
* Endgeräte: sind kleine Karten mit einem integrierten Mikrocontroller bieten Verarbeitungs- und Kommunikationsmöglichkeiten für Sensoren und Aktoren.

Die Netzwerkschicht enthält:

* Kommunikationsprotokolle: werden für Endgeräte verwendet.
* Gateway-Station: um den Informationsfluss zwischen zu steuern Endgeräte und das Internet.

Die Anwendungsebene enthält:

* IoT-Cloud-Plattformen. Die sind virtuelle Online-Datenbanken, die die Informationen vom Endgerät speichern und diese Informationen für Endbenutzer visualisieren (Tabellen, Grafiken).
* Die Software-Anwendung: für Smartphones, Tablets, Desktops mit grafischen Oberflächen zur Überwachung und Steuerung von Endgeräten.

Abbildung 2: Die Komponenten eines IoT-Modells

**2.3. WLAN**

**2.4. MQTT**

**2.5. Raspberry Pi**

**2.6. ESP8266**

**2.7. DHT11 Temperatur / Luftfeuchte Sensor**

**2.8. HC SR04 Ultraschall Sensor**

**2.9. FT232 USB-TTL Adapter**

**3. Umsetzung**

Vom ESP8266 gibt es verschiedene Modelle. Der günstigste, ESP8266 01, mit dem wir hier arbeiten, kommt mit acht Pin Rundstecker Anschlüsse, welche wiederum nur vier GPIO Pins besitzen. Dies führt auch dazu, dass der ESP nicht direkt an den Computer angeschlossen werden kann, ohne einen Adapter wie der USB-TTL Adapter einzusetzen. Die beste Lösung zu diesem Problem ist die Benutzung von Jumper Kabel um den ESP direkt an den Raspberry Pi anzuschließen.

Wir haben beide Möglichkeiten probiert, einmal mit dem Raspberry Pi, und dann auch mit dem USB-Adapter direkt am Computer. Die Programmierung vom ESP am Computer mit der Arduino IDE hat sich als effektiver erwiesen, da die IDE sehr langsam im Raspberry Pi läuft, und auch die Einrichtung mit viel Aufwand verbunden ist.

**3.1. MQTT-Broker im Raspberry Pi**

sudo apt install mosquitto

**3.2. ESP-Modul mit Raspberry Pi**

Der Raspberry Pi hat den Vorteil, selber Pins zu besitzen. Das bedeutet, wir können den ESP an den Pi ohne weiteres anschließen.

**3.3. ESP-Modul mit dem USB-TTL Adapter**

**3.4. Programmierung mit dem Arduino DIE**

- arduino ide installieren

- arduino ide für esp8266 vorbereiten (siehe link vom Niki)

- arduino ide-dht11: dht11 library runterladen und in arduino installieren(https://github.com/adafruit/DHT-sensor-library); pubsubclient in bibliothek verwaltung installieren und pubsubclient.h einbinden für mqtt, esp8266wifi einbinden

-arduino ide-ultrashall: esp8266wifi einbinden, pubsubclient.h einbinden für mqtt

Mqtt verbindung und weitere siehe code

**3.5. Visualisierung mit Node.js**

**4. Ergebnis/Ausblick (neue Aufgabe)**

**5. Literatur**