OPC UA Sensornetz

Johannes Horst, Manuel Zimmermann, Patrick Sabau, Saniye Ogul, Stefan Ries und Tobias Schotter

I. EINLEITUNG

Ziel dieses Projektes soll die Entwicklung eines Sensor-Netzwerks für den Heimbereich auf Basis von OPC-UA sein. Verschiedene Sensorknoten sollen mit entsprechender Sensorik und Aktorik ausgestattet sein, um bestimmte Daten des eigenen Zuhauses zu sammeln. Eine graphische Oberfläche ermöglicht dem Nutzer das Abrufen der Daten und außerdem Steuerungsfunktionalitäten, um z.B. eine Heizung oder eine Lüftungsanlage ein- bzw. auszuschalten. Hierdurch ergeben sich große Energiesparpotentiale, da man so nur lüftet bzw. heizt, wenn die Luftqualität/Temperatur dies erfordert.

OPC-UA steht hierbei für „Open Platform Communication – Unified Architecture“ und ist eine aktuelle Technologie aus dem Industrie-4.0-Umfeld. OPC-UA soll die plattformunabhängige Machine-to-Machine Kommunikation ermöglichen. Dies wird durch ein zu modellierendes Informationsmodell möglich, welches den realen Sachverhalt abbildet und von einem zentralen Server verwaltet wird. Clients können sich mit dem Server verbinden und dort relevante Informationen ablegen bzw. diese abrufen, bzw. durch Publish/Subscribe auch vom Server Daten erhalten. Die Informationen werden auf dem Server hierbei in einer Baumstruktur verwaltet. Durch die entsprechende Modellierung der Knoten des Baums, erhalten diese z.B. durch die Festlegung eigener Datentypen eine semantische Bedeutung.

II. VERWANDTE ARBEITEN

Als Inspiration für das Projekt dienen verschiedenste Smart-Home-Lösungen, die in den letzten Jahren von einer Vielzahl von Anbietern auf den Markt gebracht wurden. Durch die Vernetzung entsprechender Geräte, wie z.B. Lüftungsanlagen und Heizungsreglern in Kombination mit dazu passender Sensorik, können viele Vorgänge im Heimbereich automatisiert und digitalisiert werden. Eine Bedienoberfläche ergänzt solche Systeme häufig durch eine Ein- und Ausgabemöglichkeit, in welcher der Benutzer Einstellungen tätigen und wichtige Informationen abrufen kann.

Außerdem ergeben sich durch den intelligenten Einsatz von Technik gerade im Heimbereich enorme Energiesparpotentiale, da sinnloses Heizen so automatisch verhindert werden kann. Dies ist gerade im Hinblick auf die aktuelle Energiekrise in Europa eine sinnvolle Innovation.

III. ANFORDERUNGEN

Grundlegend lassen sich die Ziele der Projektarbeit in 5 Abschnitte untergliedern. Hierzu gehörten die Sensoren sowie deren Kommunikation (A), Backend (B) und das Frontend (C). Weitere optionale Features sind ebenfalls unter den entsprechenden Punkten zu finden, jedoch sind einige dieser auch in einem separaten Unterpunkt (D) aufgelistet. Die Anforderungen werden durch den Punkt Testabdeckung (E) abgerundet.

Zur Identifikation von Anforderungen werden zuerst die User Stories für die Projektarbeit betrachtet.

Das führt zum Ziel, das Projekt ausführlich anzupassen und den Integrationsaufwand so gering wie möglich zu halten.

A. Sensoren + Kommunikation

- Als Nutzer möchte ich die Werte von unterschiedlichen Sensoren auslesen können
- Als Entwickler möchte ich Aktoren anpassen und personalisieren können

B. Backend

- Als Entwickler möchte ich ein Backend, welche mittels OPC-UA empfangene Sensordaten in einer Datenbank speichert.
- Als Entwickler möchte ich die Daten für einen spezifischen Zeitraum über eine REST-API abfragen
- Als Nutzer möchte ich das Verhalten der Sensoren und Aktuatoren steuern können. Dies kann bspw. über Schwellwerte, oder aber auch durch direkte Befehle möglich sein.

C. Frontend

- Als Nutzer möchte ich die Daten eines einzelnen Sensorknotens visualisiert bekommen
- Als Nutzer möchte ich die diversen Aktoren der Sensorknoten steuern können
- Als Nutzer möchte ich die Sensorknoten in Gruppen einteilen können
- Als Nutzer möchte ich eine Übersicht über alle Sensoren in einer Gruppe haben
- Als Nutzer möchte ich alle Aktoren einer Gruppe bedienen können

D. optionale Ziele

- *Optional:* Als Nutzer möchte ich die Funktionalitäten durch ein persönliches Konto schützen können
- *Optional:* Als Nutzer möchte ich Knoten hinzufügen/ löschen können
- *Optional:* Als Nutzer möchte ich, dass meine Daten nur verschlüsselt übertragen werden

E. Testabdeckung

Als Entwickler möchte ich eine ausreichende Testabdeckung, damit Fehler frühzeitig erkannt werden. Akzeptanzkriterien sind:

- Die Code-Qualität jeder Komponente wird durch Unit-Tests gewährleistet
- Die Code-Coverage liegt bei mindestens 50%.

IV. ARCHITEKTUR

Teile der Architektur wurden bereits in Abschnitt III. erwähnt, jedoch werden diese nochmal detaillierter spezifiziert.

A. Sensoren + Kommunikation

Als Sensorknoten werden Raspberry Pi 3 eingesetzt. Allerdings verfügen diese über keine analogen Eingänge, was das Auslesen von rein analogen Sensoren wie z.B. Luftqualitäts- und Erdfeuchtesensoren standardmäßig nicht ermöglicht. Hier werden noch, auf einer aufsteckbaren Platine aufgebrachte, passende Analog-Digital-Wandler vorgeschalten.

Auf dem Raspberry Pi wird eine Python-Anwendung zum zyklischen Sammeln der Sensordaten und zum Übertragen der Sensordaten an das Backend mittels OPC-UA eingesetzt. Steuerungsanfragen für die Aktoren aus dem Backend werden über die Publish/Subscribe-Mechanismen von OPC-UA azyklisch an die Sensorknoten übergeben.

B. Backend

Das Backend ist eine Python Anwendung, welche in regelmäßigen Abständen die Sensordaten mittels OPC-UA sammelt. Diese Daten werden anschließend in einer Datenbank gespeichert, was bsp. einen zeitlichen Vergleich ermöglicht.

Neben dem Sammeln der Daten soll es außerdem möglich sein, Aktuatoren zu steuern. Beispiel hierfür wäre eine optische Anzeige mittels LED, sofern ein definierter Schwellwert überschritten wurde.

Zusätzlich können die Daten per REST-API vom Frontend angefordert werden. Neben der Wahl der Sensoren, kann obendrein noch ein Zeitraum mit angegeben werden. Dadurch kann jeder Client spezifisch wählen, welche Daten analysiert werden sollen. Nach dem Analysieren, soll der Nutzer ergänzend noch Aktuatoren beeinflussen können, indem bspw. ein Schwellwert angepasst wird.

C. Frontend

Für die Entwicklung am Frontend hat sich Angular als geeignet erwiesen. Da in der Gruppe bereits Erfahrung in diesem vorhanden ist. Ngx-admin wird hierbei als Dashboard Template verwendet. Es wurde sich für dieses Template entschieden, wegen der modularen Strukturierung, der umfangreichen Dokumentation und der schönen optischen Oberfläche. Ngx-admin selbst verwendet die Nebular UI Bibliothek für seine Module.

Abbildung 1 zeigt eine Konzeptzeichnung des Frontends. Bei der Darstellung sollen die einzelnen Sensorknoten links als Menüpunkte im Navigator angezeigt werden. Sollten die

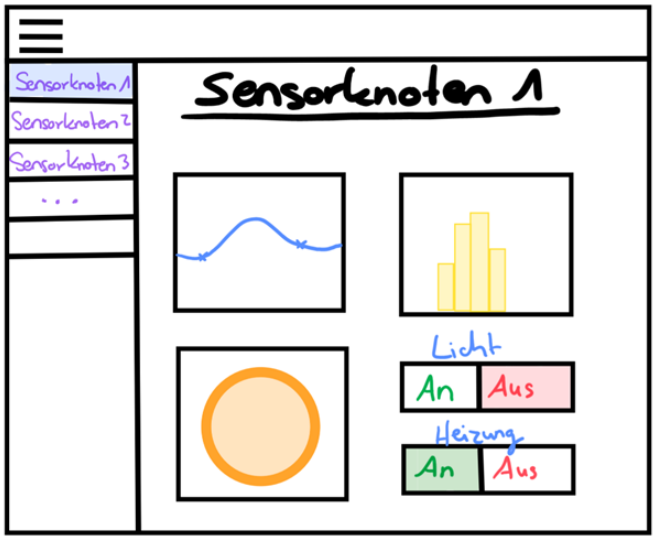

Fig. 1. Konzeptzeichnung Frontend

Knoten einer Gruppe untergeordnet sein, so ist diese im Navigator erkennbar und der Knoten steht unter einem Gruppen-Menüpunkt. Auf der rechten Seite soll das eigentliche Dashboard dargestellt werden. Hier befinden sich zum einen Diagramme zur Visualisierung/ Auswertung der Daten und zum anderen Schaltflächen für das Bedienen von Aktoren. Bei der Auswahl eines einzelnen Knotens sollen die Daten dieses speziellen Knotens visualisiert werden, bei der Auswahl einer Gruppe soll ein Überblick über die Gruppe gezeigt werden.

D. Gesamtarchitektur

Abbildung 2 zeigt einen symbolischen Aufbau der Anwendung.

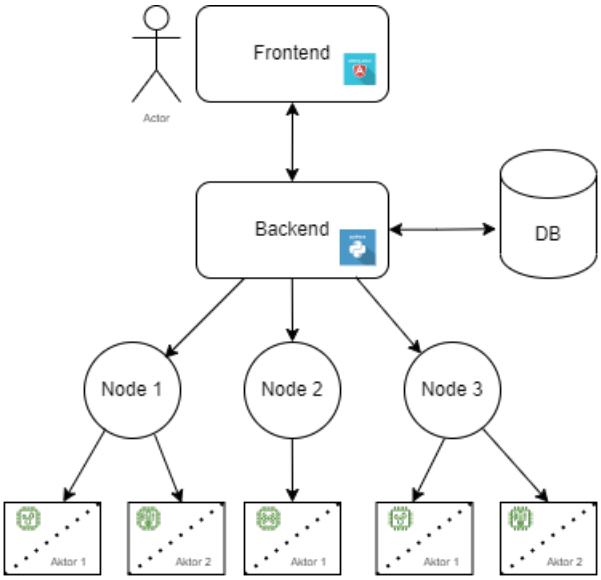

Fig. 2. Konzept Architektur