Pflichtenheft

Energy Management System

Projektteam

Karrer Alexander

Kolberger Marco

Datum: 9.5.2019

Version: 1.0

1 Einleitung **3**
1.1 Zweck des Dokuments **3**
1.2 Begriffsbestimmungen und Abkürzungen **3**
1.3 Überblick über das Dokument **3**
1.4 Zeitlicher Realisierungsrahmen **3**
2 Allgemeine Beschreibung des Produkts **4**
2.1 Zweck des Produkts **4**
2.2.1 Musskriterien (Karrer) 4
2.2.1 Musskriterien (Kolberger) 4
2.2.2 Wunschkriterien (Karrer) 4
2.2.2 Wunschkriterien (Kolberger) 4
2.2.3 Abgrenzungskriterien (Karrer) 5
2.3 Allgemeine Einschränkungen 5
Der Energy Management System Server soll auf einem Raspberry Pi 3 stabil laufen. 5
2.5 Benutzer des Produkts **5**
3 Detaillierte Beschreibung der geforderten Produktmerkmale **5**
3.1 Lieferumfang **6**
3.2 Abläufe (Szenarien) von Interaktionen **6**
3.3 Ziele des Benutzers **6**
3.4 Geforderte Funktionen des Produkts **6**
3.4.1 Auslesen der Energiedaten (Karrer) – F1 **6**
3.4.1.1 Aufruf **6**
3.4.1.2 Wirkungsweise von F1 **6**
3.4.1.3 Abhängigkeiten / Randbedingungen **7**
3.4.2 REST API anbieten (Karrer) – F2 **7**
3.4.2.1 Aufruf **7**
3.4.2.2 Wirkungsweise von F2 **7**
3.4.2.3 Abhängigkeiten / Randbedingungen **7**
3.4.3 Daten Visualisieren (Karrer) – F3 **7**
3.4.3.1 Aufruf **7**
3.4.3.2 Wirkungsweise von F3 **7**
3.4.3.3 Abhängigkeiten / Randbedingungen **7**
3.4.4 Auslesen der Energiedaten (Kolberger) - F4 **7**
3.4.4.1 Wirkungsweise von F4 **7**
3.4.5 Daten verarbeiten -F5 **7**
3.4.5.1 Wirkungsweise von F5 8
3.6 Externe Schnittstellen des Produkts **8**
3.6.1 Benutzerschnittstellen (User Interfaces) (Karrer) 8
3.6.2 Systemschnittstellen (Modbus) (Kolberger) 8
Der Raspberry Pi wird über Modbus kommunizieren. 8

3.5 Sonstige geforderte Produktmerkmale **8**
3.5.1 Geschwindigkeitsmerkmale (performance) 8
3.5.2 Schutzmerkmale (security) (Kolberger) 8
3.5.3 Sicherheitsmerkmale (safety) 8
3.5.4 Portabilitätsmerkmale (portability) 8
3.5.6 Benutzbarkeitsmerkmale (usability) 8

4 Vorgaben an die Projektabwicklung **8**
4.1 Anforderungen an die Realisierung **8**
4.2 Fertige und zugekaufte Komponenten **9**
4.3 Abnahmebedingungen **9**
4.4 Lieferbedingungen **9**

1 Einleitung

1.1 Zweck des Dokuments

Der Zweck des vorliegenden Pflichtenhefts ist eine für die Entwicklung verbindliche und möglichst eindeutige Spezifikation des Energy Management Systems, der Schnittstellen zu diesem und der Android Anwendung. In diesem Sinn enthält es die Summe aller aus Projektsicht erforderlichen und akzeptierten Anforderungen an dieses Produkt und die Projektabwicklung.

1.2 Begriffsbestimmungen und Abkürzungen

Auflistung von Definitionen und Abkürzungen, auch Begriffe aus der Domäne.

1.3 Überblick über das Dokument

Das Dokument (Pflichtenheft) enthält eine allgemeine Beschreibung des Produkts, welche unter anderem einen Überblick über dessen geforderte Funktionalität, allgemeine Einschränkungen und Vorgaben zur Hard -/ Software klassifiziert.

Ebenso beinhaltet es eine detailliertere Beschreibung der Produktmerkmale: Geschwindigkeit, Ressourcenbedarf, Portabilität, Sicherheit, Benutzbarkeit.

1.4 Zeitlicher Realisierungsrahmen

Projektvorschlag Einreichung: 9. April 2019
Genehmigung: 10. April 2019
Abgabe: 1. Juli 2019
Präsentation: 1. Juli 2019

2 Allgemeine Beschreibung des Produkts

2.1 Zweck des Produkts

Das Produkt hat die geforderte Funktionalität, welche im Anschluss behandelt wird, um Informationsgewinnung zu gewährleisten und eine Auswertung des Energieverbrauchs zu erstellen. (Karrer)

Das Projekt setzt sich als Ziel, eine Einfache Business Logic auf einem Raspberry Pi zu integrieren. Die Daten werden von einem Wechselstromrichter und einer Batterie geliefert, diese werden genutzt um eine “smarte” Aktion durchzuführen, welche von den erhaltenen Daten abhängig ist. (Kolberger)

2.2 Überblick über die geforderte Funktionalität

2.2.1 Musskriterien (Karrer)

- Erstellung eines Energy Management Systems
    - Auslesen des Energieverbrauchs eines Aktors über KNX
    - Speichern der Daten in einer Datenbank
    - Anbieten der Daten über eine REST API
- Visualisieren der Daten über eine Android App

2.2.1 Musskriterien (Kolberger)

- Erstellung eines Energy Management Systems
    - Auslesen der Stromerzeugung und des Ladezustands einer Batterie
    - Ausgabe der Erhaltenen Werte durch farbliche LED
- Das System kann die erhaltenen Werte betrachten und daraus eine “smarte” Entscheidung abgeben

2.2.2 Wunschkriterien (Karrer)

- Anleitung zur Installation des Systems
- Erweiterung der Visualisierung durch Raumtrennung und Gruppierung

2.2.3 Abgrenzungskriterien (Karrer)

- Steuerung von Aktoren aufgrund von Daten
- Auslesen von nicht KNX Aktoren

2.3 Allgemeine Einschränkungen

Der Energy Management System Server soll auf einem Raspberry Pi 3 stabil laufen.

Die App zur Visualisierung wird ausschließlich für Android Geräte mit API Level 24 (7.0) oder höher in der Sprache Englisch entwickelt. Die App kann nur über die mitgelieferte APK installiert werden und ist nicht im App-Store verfügbar. Dafür muss in den Einstellungen “Apps aus fremden Quellen zulassen” aktiviert sein. (Karrer)

Der Raspberry Pi muss mit Strom versorgt werden. Die Energieerzeuger sind über Modbus RTU mit dem Raspberry Pi verbunden, es wird ein Master / Slave System implementiert. (Kolberger)

2.4 Vorgaben zu Hardware und Software

Die entwickelte Software läuft auf jedem Android fähigen Gerät mit der Version 7.0 (API 24) des Android Betriebssystems.
Der Server läuft nur auf einem Raspberry Pi 3. (Karrer)
Der Raspberry Pi wird mit Java programmiert. Die Batterie der Firma Fronius wird über Modbus RTU angesprochen. (Kolberger)

2.5 Benutzer des Produkts

Die Verwendung des Systems erfordert Einsteiger Kenntnisse in der Home Automation, um den Server aufzusetzen.
Die Android App erfordert keinerlei Vorkenntnisse.
Die Verwendung der Software ist mit jeder Altersklasse möglich. Empfohlen ist die App für Personen mit einem technischen Grundverständnis. (Karrer)
Das Projekt ist speziell für das Energy-Lab an der FH Hagenberg konzipiert. Ein freier Zugang für andere Benutzer ist nicht vorgesehen. (Kolberger)

3 Detaillierte Beschreibung der geforderten Produktmerkmale

- Auslesen des Energieverbrauchs eines Aktors über KNX (Karrer)
    - Auslesen des aktuellen Stromverbrauchs und etwaiger Metadaten wie Uhrzeit, Aktoridentifikation, Aktortyp, etc…
- Speichern der Daten in einer Datenbank (Karrer)
    - Umwandeln des Inputs der Aktoren in vordefinierte Datenbankeinträge.
- Anbieten der Daten über eine REST API (Karrer)
    - Bereitstellung der Datenbankdaten über eine REST API mit nodejs.

- Visualisieren der Daten über eine Android App (Karrer)
    - Visualisierung der Energieverbauchsdaten aus der Datenbank mithilfe einer Android App.
- Der Benutzer kann Energieerzeuger hinzufügen und entfernen (Kolberger)
- Der Benutzer kann durch eine Ausgabe erkennen ob eine Batterie entladen oder geladen wird. (Kolberger)
- Das System wird durch eingehende Daten dazu in der Lage sein eine Entscheidung zu treffen (Kolberger)
    - Stromverbraucher aus- oder einzuschalten

3.1 Lieferumfang

Das Produkt umfasst den Raspberry Pi, welcher für die Logik zuständig ist. Die dazu ausgearbeitete Dokumentation welche aus Pflichtenheft, Meeting Mitschriften, Timetable, User Guideline, Systemarchitektur, Präsentation und Postern besteht.

3.2 Abläufe (Szenarien) von Interaktionen

1. User öffnet die App.
2. Die App erfragt die Daten des eingestellten Zeitraumes.
3. Die App visualisiert die abgefragten Daten mithilfe von Diagrammen.

3.3 Ziele des Benutzers

Das Ziel des Benutzers besteht darin, durch das Bewegen in einem Bereich die Position mit der besten Signalqualität zu Ermitteln.

3.4 Geforderte Funktionen des Produkts

3.4.1 Auslesen der Energiedaten (Karrer) – F1

3.4.1.1 Aufruf

Kontinuierliche automatische Abfrage nach einem Intervall.

3.4.1.2 Wirkungsweise von F1

Fragt jeden der Aktoren nach seinem aktuellen Stromverbrauch und speichert diesen in der Datenbank.

3.4.1.3 Abhängigkeiten / Randbedingungen

Aktor Adressen müssen bekannt sein.

3.4.2 REST API anbieten (Karrer) – F2

3.4.2.1 Aufruf

Die REST API ist dauerhaft verfügbar und kann jederzeit aufgerufen werden. Sie wird bei Serverstart gestartet.

3.4.2.2 Wirkungsweise von F2

Stellt ein Interface zur Verfügung, über welches die Android App die Daten aus der Datenbank verwerten kann.

3.4.2.3 Abhängigkeiten / Randbedingungen

Der Server muss laufen.

3.4.3 Daten Visualisieren (Karrer) – F3

3.4.3.1 Aufruf

Starten der Android Anwendung.

3.4.3.2 Wirkungsweise von F3

Durch öffnen der Android App werden Abfragen auf den Server gesendet (siehe F2) um die Daten danach visualisieren zu können.

3.4.3.3 Abhängigkeiten / Randbedingungen

Das Mobile Gerät muss im gleichen Netzwerk wie der Server sein.

3.4.4 Auslesen der Energiedaten (Kolberger) - F4

Kontinuierliche Abfrage über den Zustand der verbundenen Stromerzeuger.

3.4.4.1 Wirkungsweise von F4

Das System fragt über Modbus RTU jeden verbunden Stromerzeuger nach seinem derzeitigen Zustand.

3.4.5 Daten verarbeiten (Kolberger) - F5

Die eingehenden Daten von F4 werden analysiert und infolge dessen wird versucht eine “smarte” Entscheidung für die Stromverbraucher zu fällen.

3.4.5.1 Wirkungsweise von F5

Das System schaltet Stromverbraucher ab oder an.

3.6 Externe Schnittstellen des Produkts

3.6.1 Benutzerschnittstellen (User Interfaces) (Karrer)

Der Benutzer interagiert mit dem System nur über die Android App um sich die Visualisierung anzusehen.

3.6.2 Systemschnittstellen (Modbus) (Kolberger)

Der Raspberry Pi wird über Modbus RTU mit den Systemen von Fronius kommunizieren. Dabei wird der Raspberry Pi der Master der Master/Slave Architektur sein. Ein Modbus RTU Frame ist folgendermaßen Aufgebaut:

- mindestens 3,5 char am Anfang und Ende des Frames, da Modbus RTU stream basiert ist und so die Nachrichten unterschieden werden.
- 8 Adressen Bits, wo die ID des Slaves steht. Dadurch kann ein Master bis zu 247 Slaves ansteuern.
- 8 Function Bits, welche Unterschiedliche Read / Write Methoden aufrufen können.
- n * 8 Bits, hier stehen die übermittelten Daten
- 16 Bits CRC Check, welcher eine Prüfsumme erstellt und woraus gelesen werden kann, ob Fehler bei der Übertragung passiert sind.
- 3,5 char Pause bis zum nächsten Frame

3.5 Sonstige geforderte Produktmerkmale

3.5.1 Geschwindigkeitsmerkmale (performance)

Das System muss in der Lage sein die Energiedaten schneller zu verarbeiten, als es Daten bekommt.

3.5.2 Schutzmerkmale (security)

Es handelt sich bei dem Produkt um einen Prototypen. Es sind keine Schutzvorkehrungen implementiert, da die Funktionsweise des Systems im Vordergrund steht. (Kolberger)
Da die App eine Verbindung zum Server benötigt muss die Kommunikation zwischen App und Server verschlüsselt erfolgen (HTTPS). Dies wird allerdings im Rahmen des Projektes nicht realisiert, da das Projekt nur als Prototyp realisiert wird. (Karrer)

3.5.4 Portabilitätsmerkmale (portability)

Auf allen Android Geräten mit der Version 7.0 oder höher installierbar. Server läuft auf dem Raspberry Pi 3.

3.5.6 Benutzbarkeitsmerkmale (usability)

Alle Funktionen der App werden über Fingerberührungen gesteuert. Die Serveranwendungen müssen nach dem Setup nicht mehr vom User gesteuert werden.

4 Vorgaben an die Projektabwicklung

4.1 Anforderungen an die Realisierung

- Hardware
    - Android fähiges Gerät
    - Raspberry Pi 3
- Software
    - Windows
    - Android Studio
    - Git
    - Node.js
    - MongoDB

4.2 Fertige und zugekaufte Komponenten

Die benötigte Hardware wird von der FH Hagenberg bereitgestellt. (Kolberger)

4.3 Abnahmebedingungen

Das Projekt muss bis spätestens 5. Juli 2019 auf den dafür vorgegebenen SVN Server geladen sein. Die Dokumente gelten als abgegeben, wenn die Beispiele und Vorgabe Dokumente vom Projektteam durch ausgearbeitete Dokumente ersetzt werden.

4.4 Lieferbedingungen

Das Projekt wird zum vorgegebenen Abgabezeitpunkt zur Gänze und mit vollständiger Dokumentation auf dem dafür vorgesehenen SVN-Server sein.