HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

IFS INSTITUTE FOR SOFTWARE

Smartphone Applikation für intelligente, induktive Heizsysteme in Grossküchen

Bachelorarbeit

HSR—Hochschule für Technik Rapperswil
Institut für Software

Herbstsemester 2015

Autoren:	Konstantin Kayed, Theodor Winter
Betreuer:	Prof. Dr. Farhad Mehta
Experte:	Vikram Kriplaney
Gegenleser:	Prof. Hansjörg Huser

Abstract

Die Firma Fluxron Solutions AG entwickelt in Amriswil Heizlösungen und Küchengeräte auf Induktionsbasis. Diese Geräte besitzen eine Bluetooth-Schnittstelle, über welche Einstellungen angepasst und ausführliche Laufzeit- und Fehlerprotokolle ausgelesen werden können. Servicetechniker benötigen genau diese Informationen zur Reparatur der Geräte in Grossküchen. Aufgrund der grossen Geräteanzahl, ist es schwierig die Installationen im Überblick zu behalten.

In dieser Arbeit wurde eine Applikation für Android entwickelt, welche Techniker zur Diagnose und Konfiguration nutzen. Die Lage der eingebauten Geräte wird auf Situationsfotos markiert. Bei einem späteren Serviceeinsatz werden diese Positionen und der Status aller Kochinstallationen abgerufen.

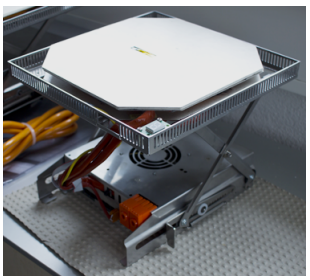
Zur Umsetzung des Projektes wurden agile Softwareentwicklungsmethoden eingesetzt. Neben einer gründlichen Anforderungsanalyse wurde die Benutzeroberfläche mit Mockups im Material Design konzipiert und mittels Usability-Walkthrough validiert.

Als Programmiersprache wurde Java 7 für Android eingesetzt. Die Anwendungsarchitektur besteht aus drei Layern, welche mittels Messages über ein Event Bus System kommunizieren. Lokal werden die Küchendaten in einer dokumentbasierten Datenbank gespeichert. Die Kommunikation mit den Geräten erfolgt über das CANopen-Protokoll. Zudem wurde die Architektur darauf ausgelegt, die Erweiterung um ein Cloud-Backend einfach zu machen.

Der Funktionsumfang der Mobilapplikation umfasst die Verwaltung mehrerer Küchen und der darin installierten Geräte. Küchen werden in einzelne Bereiche unterteilt und der Status der Geräte wird regelmässig aktualisiert. Der Funktionsumfang wurde mit einem erfolgreichen Praxistest vor Ort überprüft. Die Servicetechniker profitieren nun von einer modernen Applikation, welche ihnen den Wartungsalltag erleichtert.

Management Summary

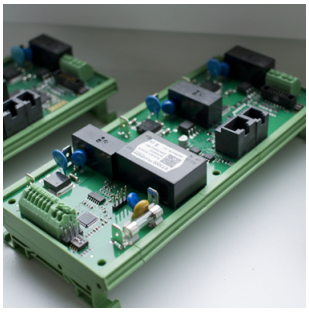
Ausgangslage



Die Firma Fluxron Solutions AG entwickelt und produziert in Amriswil Heizlösungen und Küchengeräte mit Induktionstechnologie. Die Küchengeräte sind allerdings keine Produkte für die private Küche zu Hause, sondern Einbaugeräte für Kombinationen in Grossküchen. Darunter sind neben verschiedenen Induktionsherdmodellen auch elektronisch gesteuerte Thermostate. Meistens werden die Geräte nicht durch Fluxron selbst verbaut, sondern über eine Servicefirma installiert und dann beim Endkunden in der Grossküche eingebaut.

Fluxron stattet die Geräte serienmässig mit einem Bluetoothmodul aus. Darüber können ohne Kabelverbindung die Geräteeinstellungen angepasst und ausgelesen werden. Zudem liefert die Schnittstelle auch ausführliche Laufzeit- und Fehlerprotokolle.

Ein Servicetechniker benötigt genau diese Informationen, um vor einer allfälligen Reparatur der Geräte in Grossküchen eine Diagnose des Problems zu stellen. Dies ist besonders nützlich, da häufig keine Reparatur des Gerätes nötig ist, sondern nur eine Einstellung geändert werden muss. Oft wird eine Vielzahl von Geräten verbaut und es ist für die Techniker schwierig, die Installationen auch bei einem späteren Serviceeinsatz noch genau zu kennen. Sie sollen daher durch eine Smartphone Applikation unterstützt werden.



Die Firma hat bereits eine solche Applikation im Einsatz, allerdings kann diese nur für die Verbindung mit einem einzelnen Gerät genutzt werden. Daher soll eine neue App

entwickelt werden. Die Techniker sollen die Installationen in den verschiedenen Küchen speichern und bei der Wartung wieder abrufen können. Sie sollen die Geräte in einem Suchlauf finden, deren Position in der Küche markieren und den Status einsehen können. Zudem soll es möglich sein, die Einstellungen der Geräte mit der App anzupassen.

Vorgehen

Zur Durchführung des Projektes wurden agile Methoden der Softwareentwicklung angewendet. Initialisiert wurde das Projekt mit einer gründlichen Anforderungsanalyse mittels User Stories und Use Cases. In wöchentlichen Meetings wurden die weiteren Aufgaben priorisiert und in planbare Tasks aufgeteilt.

Aufbauend auf den Anforderungen wurde dann die Benutzeroberfläche mittels Mockups entworfen und im Walkthrough mit dem Kunden besprochen. Dies lieferte die Grundlage für die technische Konzeption der Applikation, welche aus Architekturdefinition und Evaluationen zur genutzten Technologie besteht.

Nach Analyse und Konzeption wurde die Anwendung iterativ im Wochenrhythmus implementiert und getestet. Um die korrekte Funktionsweise zu garantieren, wurde die App abschliessend nochmals vor Ort mit vollständigen Geräteinstallationen erfolgreich getestet.

Technologien

Die Anwendung wurde mit Java 7 für Android umgesetzt. Um eine gute Kompatibilität zu gewährleisten, ist die App ab Androidversion 4.3 und höher nutzbar. Die Anwendungsarchitektur besteht aus drei Ebenen, welche mittels Meldungen über den Green Robot Event Bus kommunizieren.

Alle Anwendungsdaten werden in einer lokalen Couchbase Lite Datenbank gespeichert und können über die Exportfunktion per EMail ausgetauscht werden. Die Kommunikation mit der Hardware erfolgt über das CANopen-Protokoll und funktioniert somit für alle Küchengeräte der Firma Fluxron.

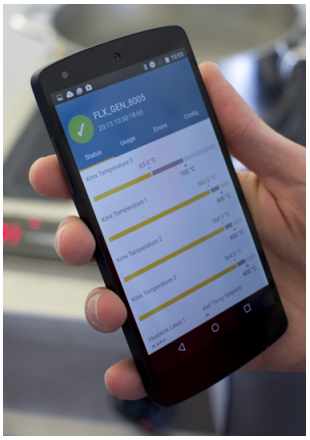
Ergebnisse

Die Anwendung unterstützt das Erfassen einer beliebigen Anzahl von Küchen. Der Techniker kann für jede Installation eine eigene Küche erfassen und diese einfach über die Suchfunktion wieder finden. Küchen können beliebig mit zusätzlichen Notizen ergänzt werden.

Um die Situation in einer Grossküche genau erfassen zu können, kann der Techniker die Küche in verschiedene Bereiche unterteilen. Diese werden jeweils mit einem Foto hinterlegt und sind so einfach und effizient erfasst.

Damit die Servicetechniker die genaue Position der eingebauten Geräte auch später bei einem Wartungsauftrag noch kennen, können die aktiven Geräte mit einem Suchlauf aufgelistet und auf einem Küchenbereich platziert werden.

Für Servicezwecke kann nun ohne Ausbau oder direkten Zugang zu den Geräten der Status abgefragt werden. Der Techniker sieht auf einen Blick alle Sensorwerte und kann sich die Fehlerprotokolle und Statuszähler anzeigen lassen.



Die Werte werden durch die Applikation für alle Geräte automatisch aktualisiert und ermöglichen somit ein effizientes Arbeiten. Auch bei der Diagnose von Problemen bei mehreren Geräten ist nur wenig manuelle Interaktion notwendig.

Damit wird das Problem der umständlichen Diagnose und Wartung der Geräte gelöst. Die Techniker erfassen die Situation der Küche bereits bei der Installation und können diese dann bei der Wartung wieder abrufen. Dabei ist kein manueller Verbindungsaufbau mehr nötig, da die Geräte bereits erfasst sind.

Ausblick

Die Anwendung wird an die Firma Fluxron übergeben und dort zur Marktreife weiterentwickelt um von Servicefirmen und Technikern der Firma Fluxron verwendet zu werden.

In der aktuellen Version der App wurden aus Zeitgründen nur die Gerätetypen C- und S-Klasse vollständig angebunden. Es ist aber problemlos möglich, weitere Fluxron-Geräte anzusprechen. Die Anwendung unterscheidet die Geräte bereits an ihrem Herstellercode und kann daher um beliebige Gerätetypen erweitert werden. Es muss lediglich die Benutzeroberfläche für die entsprechenden Gerätetypen definiert werden. Neben der Oberfläche kann auch ein neuer Parametersatz (EDS-Datei) integriert werden. Dieser wird dann über die automatische Code-Generierung für den Entwickler vorbereitet und kann einfach mit der Benutzeroberfläche verbunden werden.

Nebst der Erweiterung um neue Gerätetypen bietet die Anwendung noch weiteres Potenzial für die Zukunft. So könnte zum Beispiel eine Funktion zum Einlesen und Versenden der gesamten Gerätekonfiguration neue Möglichkeiten bei Diagnose und Installation bieten.

Ausserdem ist die App bereits jetzt darauf vorbereitet, an die Cloud angebunden zu werden. Dies einerseits durch die Verwendung einer entsprechenden Datenbanktechnologie und andererseits durch die meldungsbasierte Architektur. Damit würden sich neue Möglichkeiten wie z.B. Nutzungs- oder Fehlerprotokolle bieten. Zudem könnten die Servicefirmen damit alle Küchen zentral speichern und müssten die Küchen nicht mehr per EMail verteilen.

Erklärung der Eigenständigkeit

Hiermit erklären wir,

- dass wir die vorliegende Arbeit selber und ohne fremde Hilfe durchgeführt haben, ausser derjenigen, welche explizit in der Aufgabenstellung erwähnt sind oder was mit dem Betreuer schriftlich vereinbart wurde,
- dass wir sämtliche verwendeten Quellen erwähnt und gemäss gängigen wissenschaftlichen Zitierregeln korrekt angegeben haben und,
- dass wir keine durch Copyright geschützten Materialien (z.B. Bilder) in dieser Arbeit in unerlaubter Weise genutzt haben.

Rapperswil-Jona, 16. Dezember 2015

Konstantin Kayed

Theodor Winter

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

Aufgabenstellung Bachelorarbeit
„Smartphone Applikation für intelligente, induktive Heizsysteme in Grossküchen“
HS 2015

1. Auftraggeber und Betreuer

- *Auftraggeber:* FLUXRON Solutions AG, Weinfelderstrasse 82, CH-8580 Amriswil
- *Ansprechpartner Auftraggeber:* Hr. Mischa Reichard
- *Betreuer:* Prof. Dr. Farhad Mehta

2. Studierende

Diese Arbeit wird als Bachelorarbeit an der Abteilung Informatik durchgeführt von

- Hr. Konstantin Kayed
- Hr. Theo Winter

3. Ausgangslage

Die Fluxron Solutions AG ist ein junges, innovatives Schweizer Unternehmen mit Sitz in Amriswil. Die Haupttätigkeiten sind die Entwicklung von induktiven und elektronisch geregelten Heizsystemen für die Grossküche sowie für industrielle Anwendungen. Zu den Kunden gehören europaweit namhafte Küchenbauer, welche die innovativen und technologisch hochstehenden Produkte sehr schätzen.

Fluxron Produkte sind unter anderem mit einer einem Bluetooth Modul ausgestattet und können so mit Smartphones oder anderen Bluetoothfähigen Geräten kommunizieren. In einer Grossküche sind jeweils mehrere (teilweise bis zu 100) induktive Heizsysteme mit je einem integrierten Bluetooth Modul verbaut. Konfigurationen, Optimierungen sowie Errorhandling und Serviceeinsätze der Geräte werden vorwiegend drahtlos über die Bluetooth Schnittstelle ausgeführt. Somit wird ein aufwendiges und zeitraubendes auseinanderbauen von Küchenmodulen oder Geräten umgangen.

4. Beschreibung der Aufgabe

Der Auftraggeber möchte eine Smartphone-Applikation mit folgenden Zielen realisieren:

1. Die Smartphone-Applikation muss von einem Reperaturfachman benutzbar sein.
2. Die Smartphone-Applikation muss mittels Bluetooth Classic mit den Heizsystemen kommunizieren können.
3. Die Smartphone-Applikation unterstützt einen Suchlauf zum Finden aller Fluxron Geräte.
4. Mit den installierten Geräten muss dann, mittels benutzerfreundlicher Bedienung, eine kundenspezifische Küchenabbildung erstellt und abgespeichert werden können.
5. Ist die Küche einmal initialisiert, müssen wählbare Parameter der Geräte dieser Küche durch Bluetooth-polling auf dem Smartphone empfangen, oder zu den Heizsystemen gesendet werden können.

6. Der Auftraggeber entwickelt gerade eine Bluetooth 4 Schnittstelle für seine Produkte. Die Smartphone-Applikation muss auch gleichzeitig mittels Bluetooth 4 mit zukünftigen Heizsystemen kommunizieren können.
7. Küchenabbildungen und Konfigurationen müssen ausserhalb der Smartphone-Applikation gespeichert und wiedergelesen werden können (z.B. import, export, synchronisation).

8. Die Smartphone-Applikation muss Geräteparameter auf eine zu erstellende Online-Datenbank oder Webapplikation ereignisorientiert und autonom übertragen können. Damit kann diese App auch als Fernwartungstool genutzt werden.

Für diese Bachelorarbeit werden die oben genannten Ziele wie folgt priorisiert:

- Priorität 1: Ziele 1-5
- Priorität 2: Ziele 6 & 7
- Priorität 3: Ziel 8

Ziel diese Bachelorarbeit ist ein Smartphone-Applikation zu entwickeln, die alle Ziele mit Priorität 1 erfüllt. Die Erreichung der restlichen Ziele (Prioritäten 2 & 3) ist erwähnenswert, aber nicht Pflicht. Es muss aber gezeigt werden, dass die Smartphone-Applikation erweiterte werden kann, um diese Ziele zu erfüllen.

5. Zur Durchführung

Mit dem Betreuer finden wöchentliche Besprechungen statt. Besprechungen mit dem Auftraggeber sind von den Studierenden nach Bedarf zu initialisieren.

Alle Besprechungen sind von den Studierenden mit einer Traktandenliste vorzubereiten, die Besprechung ist durch die Studierenden zu leiten und die Ergebnisse sind in einem Protokoll festzuhalten, das den Betreuern und dem Auftraggeber per E-Mail zugestellt wird.

Für die Durchführung der Arbeit ist ein Projektplan zu erstellen. Dabei ist auf einen kontinuierlichen und sichtbaren Arbeitsfortschritt zu achten. An Meilensteinen gemäss Projektplan sind einzelne Arbeitsresultate in vorläufigen Versionen abzugeben. Über die abgegebenen Arbeitsresultate erhalten die Studierenden ein vorläufiges Feedback. Eine definitive Beurteilung erfolgt auf Grund der am Abgabetermin abgelieferten Dokumentation.

6. Dokumentation

Über diese Arbeit ist eine Dokumentation gemäss den Richtlinien der Abteilung Informatik zu verfassen (siehe https://www.hsr.ch/Allgemeine-Infos-Diplom-Bach.4418.0.html). Die zu erstellenden Dokumente sind im Projektplan festzuhalten. Alle Dokumente sind nachzuführen, d.h. sie sollten den Stand der Arbeit bei der Abgabe in konsistenter Form dokumentieren. Alle Resultate sind vollständig auf CD/DVD in 3 Exemplaren abzugeben. Der Bericht ist ausgedruckt in doppelter Ausführung abzugeben.

7. Termine

Siehe Terminplan auf https://www.hsr.ch/Termine-Diplom-Bachelor-und.5142.0.html.

8. Arbeitsumfang

Eine erfolgreiche Bachelorarbeit zählt 12 ECTS-Punkte pro Studierenden. Für 1 ECTS Punkt ist eine Arbeitsleistung von 30 Stunden budgetiert. Die verwendete Arbeitszeit muss erfasst und dokumentiert werden.

9. Beurteilung

Für die Beurteilung ist der HSR-Betreuer verantwortlich. Die Benotung erfolgt gemäss folgender Tabelle.

Gesichtspunkt	Gewicht
1. Organisation, Durchführung	1/6
2. Berichte	1/6
3. Inhalt	3/6
4. Mündliche Prüfung zur Bachelorarbeit	1/6

Im Übrigen gelten die Bestimmungen der Abteilung Informatik für Bachelorarbeiten (siehe https://www.hsr.ch/Ablaeufe-und-Regelungen-Studie.7479.0.html).

Prof. Dr. Farhad Mehta
Rapperswil, den 14.09.2015

Inhaltsverzeichnis

1. Analyse

1.1. Ausgangslage

1.1.1. Anwendungsumfeld

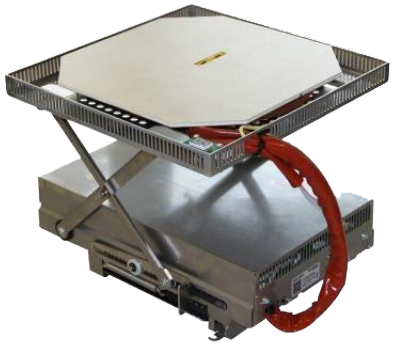


Abbildung 1.1.: Induktionsherd

Die Firma Fluxron AG mit Sitz in Amriswil TG bietet induktive Heiz- und Energiesysteme für Grossküchen an. Die Produktpalette besteht aus verschiedenen Induktionsherden und Thermostatsystemen. Diese haben jeweils ein Bluetoothmodul eingebaut, welches auf ein CANopen-basiertes Protokoll zur Kommunikation setzt. Die Module liefern neben den Geräteeinstellungen auch Fehlercodes und Sensormesswerte via Bluetooth.

Fluxron verkauft diese Induktionsgeräte an Servicefirmen, welche diese dann beim Endkunden, z.B. einem Restaurant, einbauen und installieren. Dabei, aber auch bei Wartungsarbeiten, setzen die Servicefirmen die bestehende Android Applikation „FLX Tool“ zur Diagnose und zur Konfiguration ein.

Neben den Servicefirmen setzen auch die Mitarbeiter der Firma Fluxron die Androidapplikation bei internen Versuchen oder zur Ferndiagnose ein. Bei der Ferndiagnose wird eine Teamviewer-Verbindung auf das Smartphone oder den PC des Technikers aufgebaut. Über diese kann dann eine Diagnose mittels den Tools via Bluetooth erfolgen.

1.1.2. Bestehende Android-Applikation - Fluxron Systemkonfigurator

Fluxron hat in Eigenentwicklung bereits eine einfache Android-Applikation geschrieben. Diese unterstützt das Suchen von Geräten sowie das Lesen und Schreiben von Parametern. Da diese App allerdings immer nur ein Gerät gleichzeitig bedienen kann und die Benutzerinteraktion daher sehr zeitaufwändig ist, soll im Rahmen dieses Projektes eine neue, übersichtlichere Androidapplikation entstehen. In den nachfolgenden Abschnitten sind die Funktionen dieser Applikation aufgelistet.

Installation und Konfiguration

Die App ist öffentlich im Google Play Store erhältlich. Nach dem Download muss der Benutzer aber ein Kennwort eingeben, um Zugang zur App und den Gerätefunktionen zu erhalten.

Abbildung 1.2.: Geräteauswahl FLX Tool

Verbindungsaufbau zu einem Gerät

Um eine Verbindung mit einem Gerät herzustellen, muss dieses zuerst über einen Suchlauf gefunden werden. Der Suchlauf zeigt alle momentan aktiven Geräte in einer Liste mit ihrer Device-ID an. In dieser Liste sind neben den aktiven Geräten auch alle Geräte aufgelistet, welche schon einmal mit der App verbunden waren.

Danach kann man sich mit einem Gerät aus der Liste verbinden um mit dem Gerät zu interagieren.

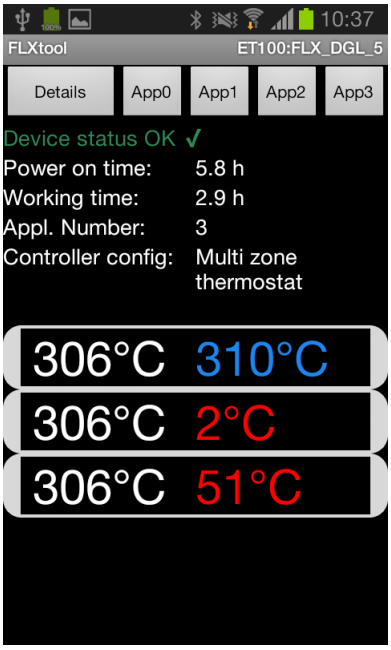


Abbildung 1.3.: Statusansicht Thermostat

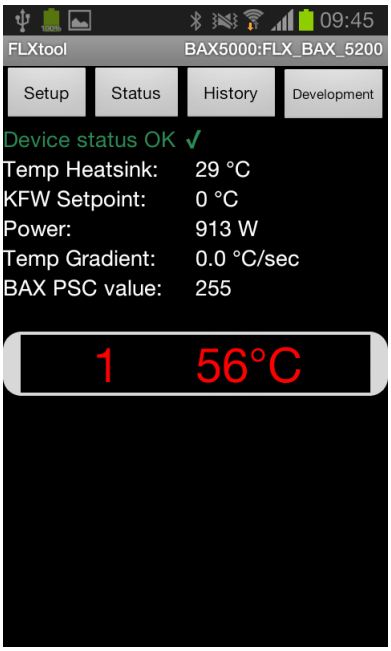


Abbildung 1.4.: Statusansicht für Induktionsherd

Setupansicht

Wenn ein Gerät verbunden ist, kann in der Setupansicht eine Liste aller Parameter des Gerätes ausgelesen werden. Der Benutzer kann Parameter anwählen, eine Beschreibung ansehen und Werte ändern.

Historyansicht

In der Historyansicht können Betriebszähler eingesehen werden. Sie enthält Werte wie die Anzahl von „Power-On“ Events oder Laufzeit des Gerätes.

Neben den Zählern kann ein Fehlerlog eingesehen werden, welches die letzten zehn Fehlercodes zeigt. Dies wird zur Diagnose von Problemen genutzt.

Einschränkungen der App

- Es kann immer nur ein Gerät gleichzeitig verbunden sein.
- Für jede Verbindung ist ein erneuter Suchlauf nötig.
- Die Geräte sind nur mit der ID sichtbar, was die Orientierung in einer Grossküche mit einem dutzend Geräten schwierig macht.
- Wenn ein nicht verbundenes Gerät eine Fehlermeldung absetzt, wird diese nicht empfangen.
- Der Typ der verbundenen Geräte wird zwar erkannt, die Benutzeroberfläche bietet aber immer noch alle Gerätetypen an.
- Wenn keine Geräte vorhanden sind, oder gefunden werden, hat die App keinen Nutzen für den User.

1.1.3. Weitere bestehende Software

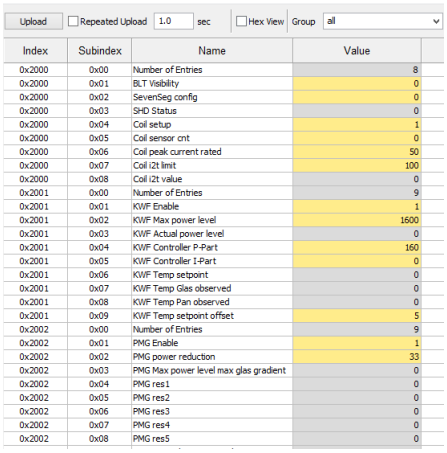


Abbildung 1.5.: FLX Access

FLX Access

Mit FLX Access stellt Fluxron ein Windows-Tool zum Auslesen und Setzen von gerätespezifischen Parametern bereit. Die Kommunikation erfolgt via Bluetooth. Zudem kann der Gerätestatus ausgelesen werden.

Dieses System ist besonders für die Entwicklungsphase von Geräten geeignet.

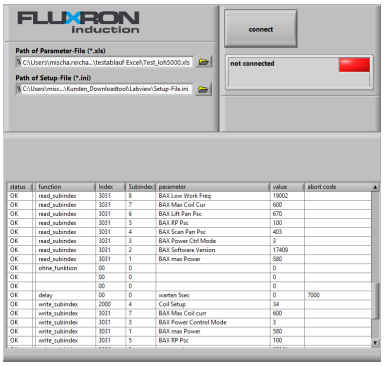


Abbildung 1.6.: FLX Downloadtool

FLX Downloadtool

Das FLX Downloadtool ermöglicht den Download der gesamten Konfiguration in eine Excel-Datei. In dieser können dann die Parameter eingestellt werden. Danach werden die Einstellungen über das Tool wieder auf das Gerät geladen.

Eingesetzt wird dieses Programm für das Schreiben von kundenspezifischen Parametern auf mehrere Geräte.

1.2. Domainanalyse

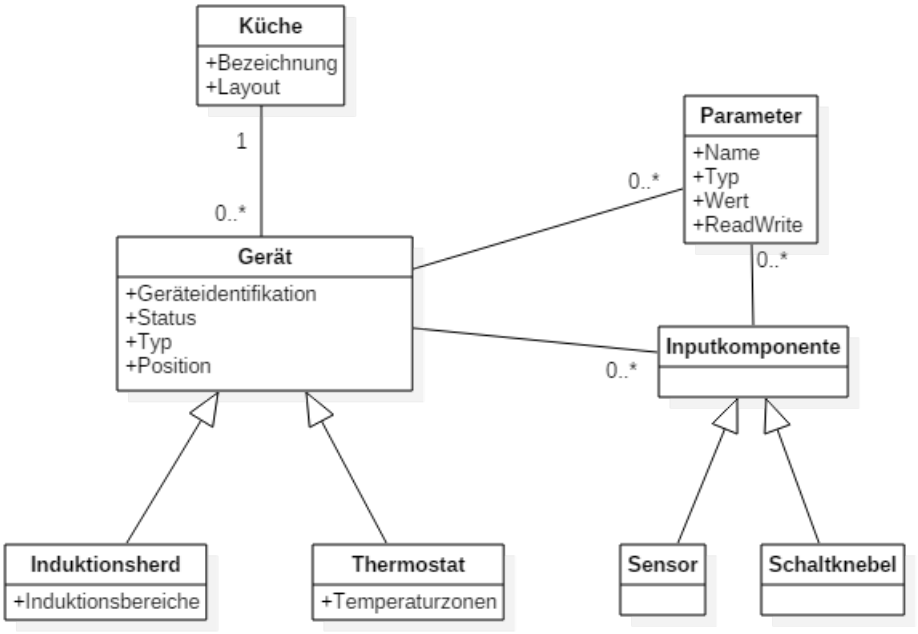


Abbildung 1.7.: Domainanalyse „Küche mit Geräten und Parametern“

1.2.1. Kurzbeschreibung der Begriffe

Küche

Eine Küche ist im Kontext dieser Applikation „nur“ eine Ansammlung von Geräten mit ihrer Positionierung. Als Layout einer Küche werden die Gegebenheiten vor Ort bezeichnet. Dies kann z.B. ein Grundriss oder ein Foto der Küche sein. Dabei werden nur die wesentlichen Elemente betrachtet, d.h. keine Spülbecken oder Abfalleimer.

Gerät

Ein Gerät wird über seine Geräteidentifikation (ID) identifiziert. Es hat einen Status und einen Typ. Zudem ist sein Einbauort in der Küche für den Benutzer wichtig.

Parameter

Die Gerätekonfiguration besteht aus einer Liste von Parametern. Diese unterscheiden sich durch ihren Namen und ihren Datentyp. Zudem gibt es Parameter, welche nur ausgelesen, aber nicht verändert werden können.

Induktionsherd

Induktionsherde erhitzen das Kochgut mittels elektromagnetischen Effekten. Dabei wird über eine Spule ein Magnetfeld erzeugt, welches die Pfanne oder den Topf direkt anregt und somit erhitzt.

Diese Kochmodule haben Temperaturfühler und eine Topferkennung. Bei der Erkennung wird sichergestellt, dass die Leistungsabgabe nur erfolgt, wenn auch ein gefüllter Topf auf der Oberfläche steht.

Die Firma Fluxron vertreibt verschiedene Modelle von Induktionsgeräten, diese unterscheiden sich in ihren Funktionalitäten. Einige Modelle bieten unterschiedlich steuerbare Kochzonen, andere haben eine höhenverstellbare Spule für einen flexibleren Einbau.

Thermostat

Die Thermostaten werden meist in andere Geräte, wie z.B. einen Kontaktgrill, verbaut. Dabei messen sie die Temperatur der Heizfläche und geben den Heizstrom frei, oder unterbrechen ihn. Auch hier gibt es verschiedene Varianten mit einer unterschiedlichen Anzahl von Heizzonen (Solltemperatur + Regelausgang).

Inputkomponente

Inputkomponenten lesen externe Werte ein. Dies kann z.B. ein Temperatursensor oder ein Schaltknebel (Drehknopf) sein. Die Sensoren und Schaltknebel werden im Hintergrund ebenfalls als Set von Parametern abgebildet.

1.3. User Stories

1.3.1. Höchste Priorität

1. Als Servicetechniker oder Mitarbeiter der Fluxron möchte ich alle, über Bluetooth erreichbaren Geräte, nach Typ gruppiert mit einem Suchlauf finden können.
2. Als Servicetechniker möchte ich eine übersichtliche Darstellung der Küche und ihrer Geräte erfassen um diese bei einem Wartungstermin nutzen zu können. Damit kann ich die Geräte leicht wiederfinden.
3. Es kann vorkommen dass manche Geräte zwischenzeitlich nicht erreichbar sind. Als Techniker möchte ich diese Geräte dennoch auf der Übersicht sehen, will aber klar erkennen, dass keine Verbindung besteht.
4. Als Servicetechniker möchte ich vor Ort Parameter und Protokolle der Geräte auslesen können. Wenn ich eine Einstellung verändern möchte, soll dies ebenfalls über die App funktionieren.
5. Als Servicetechniker möchte ich jederzeit den Überblick über den Status aller Geräte behalten können.
6. Als Entwickler von Fluxron möchte ich eine einfache Erweiterbarkeit des Programmcodes, damit ich die App später leicht anpassen kann.

1.3.2. Zweite Priorität

7. Als Mitarbeiter von Fluxron möchte ich die neue Gerätegeneration mit Bluetooth 4.0 ebenfalls ansprechen können.
8. Als angestellter Servicetechniker in einer Firma möchte ich die Küchenkonfiguration und das Layout mit meinem Mitarbeiter austauschen oder sichern können.

1.3.3. Dritte Priorität

9. Als Entwickler von Fluxron möchte ich zu einem späteren Zeitpunkt eine Internetanbindung für die App realisieren. (Konzeptionelle Unterstützung bereits jetzt in der App)

10. Als Entwickler von Fluxron möchte ich zu einem späteren Zeitpunkt automatische Fehlermeldungen von meinem Produkt über das Internet erhalten. (Konzeptionelle Unterstützung bereits jetzt in der App)
11. Als Servicetechniker möchte ich über das Internet informiert werden, wenn ein von mir installiertes Gerät einen Fehler auslöst. (Konzeptionelle Unterstützung bereits jetzt in der App)
12. Als Techniker von Fluxron möchte ich zu einem späteren Zeitpunkt automatische Protokolle zu Nutzung und Verschleiss sehen können. (Konzeptionelle Unterstützung bereits jetzt in der App)
13. Als Techniker von Fluxron möchte ich zu einem späteren Zeitpunkt automatische Protokolle zu den Parameter- und Messwerten sehen. (Konzeptionelle Unterstützung bereits jetzt in der App)

1.4. Use Cases

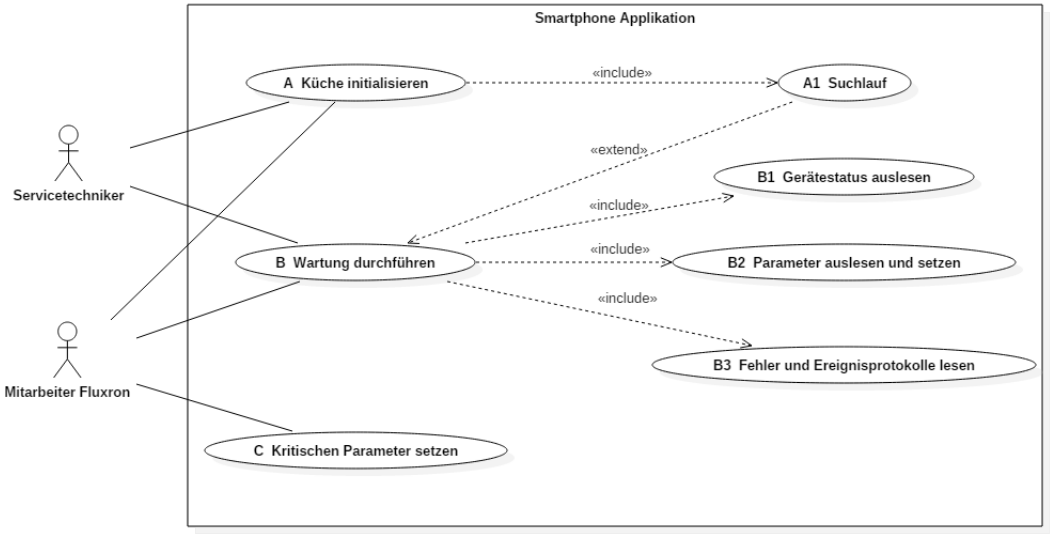


Abbildung 1.8.: Anwendungsfälle für die neue Smartphone-Applikation

1.4.1. A - Küche initialisieren

Use Case A - Küche initialisieren	
Aktoren	Servicetechniker oder Mitarbeiter Fluxron
Description	Der Aktor nutzt die App um die Lage der Geräte in der Küche zu erfassen.
Preconditions	• Die App ist gestartet • Bluetoothverbindung des Gerätes verfügbar • Passwort für die App wurde eingegeben
Postconditions	Die Küche ist nach dem Verlassen des Use Cases gespeichert
Trigger	Der Aktor startet in der App die Funktion zur Erfassung einer Küche
Normal flow	1. Der Aktor erstellt eine neue Küche und gibt dieser einen Namen 2. Der Aktor stellt das Layout der Küche in der App nach 3. Der Aktor startet den Suchlauf 4. Der Aktor platziert die gefundenen Geräte auf dem Layout
Alternative flows	
Exceptions	Der Benutzer kann diesen UseCase jederzeit verlassen und wiederaufnehmen.

Tabelle 1.1.: Use Case A - Küche initialisieren

1.4.2. B - Wartung durchführen

Use Case B - Wartung durchführen	
Aktoren	Servicetechniker oder Mitarbeiter Fluxron
Description	Der Aktor nutzt die App um sich einen Überblick über den Gerätestatus in einer bestehenden Küche zu verschaffen, er ändert Parameter und liest Protokolle.
Preconditions	• Die App ist gestartet • Bluetoothverbindung des Gerätes verfügbar • Küche ist bereits erfasst
Postconditions	Die Küche ist nach dem Verlassen des Use Cases gespeichert
Trigger	Der Aktor hat ein Problem mit einem Gerät
Normal flow	1. Der Aktor lädt die Küche a) Er kann nun alle Geräte in der Küche mit ihrem aktuellen Status sehen b) Er kann zusätzlich gefundene Geräte platzieren 2. Der Aktor wählt ein Gerät 3. Der Aktor liest die Protokollspeicher des Gerätes aus 4. Der Aktor kontrolliert die Parameterliste 5. Der Aktor ändert einen Parameter um das Problem zu beheben
Alternative flows	1a: Es wird ein Suchlauf durchgeführt, dieser dient dazu, neu installierte Geräte ebenfalls zu finden
Exceptions	Der Benutzer kann diesen UseCase jederzeit verlassen und wiederaufnehmen.

Tabelle 1.2.: Use Case B - Wartung durchführen

1.4.3. C - Kritischen Parameter setzen

Use Case C - Kritischen Parameter setzen	
Aktoren	Mitarbeiter Fluxron
Description	Der Aktor nutzt die App um einen sicherheitsrelevanten Parameter auf einem Gerät zu ändern.
Preconditions	• Die App ist gestartet • Bluetoothverbindung des Gerätes verfügbar • Küche ist bereits erfasst • Nutzer ist als Mitarbeiter von Fluxron identifiziert
Postconditions	Die Küche ist nach dem Verlassen des Use Cases gespeichert
Trigger	Der Aktor hat ein Problem mit einem Gerät oder möchte einen Versuch durchführen
Normal flow	1. Der Aktor wählt ein Gerät 2. Der Aktor ändert einen kritischen Parameter
Alternative flows	
Exceptions	

Tabelle 1.3.: Use Case C - Kritischen Parameter setzen

1.4.4. A1 - Suchlauf

Use Case A1 - Suchlauf	
Aktoren	Servicetechniker oder Mitarbeiter Fluxron
Trigger	Manueller Start oder Start durch das Laden einer Küche
Normal flow	1. Es wird im Hintergrund nach aktiven Bluetooth-Geräten gesucht 2. Die Geräte werden dem Benutzer in einer gruppierten Liste angezeigt

Tabelle 1.4.: Use Case A1 - Suchlauf

1.4.5. B1 - Gerätestatus auslesen

Use Case B1 - Gerätestatus auslesen	
Aktoren	Servicetechniker oder Mitarbeiter Fluxron
Precondition	Gerät ist gewählt
Normal flow	1. Der Aktor sieht eine Übersicht über den Gerätestatus

Tabelle 1.5.: Use Case B1 - Gerätestatus auslesen

1.4.6. B2 - Parameter auslesen und setzen

Use Case B2 - Parameter auslesen und setzen	
Aktoren	Servicetechniker oder Mitarbeiter Fluxron
Precondition	Gerät ist gewählt
Normal flow	1. Der Aktor lässt sich die Parameterliste anzeigen 2. Der Aktor wählt einen Parameter aus 3. Der Aktor gibt einen neuen Wert ein und bestätigt diesen
Exceptions	• Es wird ein, für die Masseinheit ungültiger Wert eingegeben • Es wird ein, für den Parameter ungültiger Wert (Wertebereich) eingegeben • Der Parameter kann nicht an das Gerät gesendet werden

Tabelle 1.6.: Use Case B2 - Parameter auslesen und setzen

1.4.7. B3 - Fehler- und Ereignisprotokolle auslesen

Use Case B3 - Fehler- und Ereignisprotokolle auslesen	
Aktoren	Servicetechniker oder Mitarbeiter Fluxron
Precondition	Gerät ist gewählt
Normal flow	1. Der Aktor lässt sich das Protokoll anzeigen
Exceptions	• Die Protokolle und Ereigniszähler können leer sein

Tabelle 1.7.: Use Case B3 - Fehler- und Ereignisprotokolle auslesen

1.5. Functional Requirements

Priorität 1

1. Kennwortschutz um unerlaubten Zugriffen vorzubeugen
2. Suchlauf, welcher alle gefundenen Geräte übersichtlich gruppiert darstellt
3. Layouterfassung und Verwaltung von Küchen
4. Platzieren der Geräte auf dem Küchenlayout
5. Nachträgliches Anpassen des Küchenlayouts (inkl. +/- von Geräten)
6. Auswählen einzelner Geräte aus der Layoutansicht für Details
7. Detaillierte Statusansicht für den Gerätestatus
8. Parameterliste der Geräte auslesen
9. Parameter auf einem Gerät setzen
10. Begrenzung für Masseinheit und Wertebereich der Parameter
11. Ereignisprotokolle (Zähler) und Fehlerprotokolle auslesen
12. Überblick über den Gerätestatus im Küchenlayout
13. Einfache Erweiterbarkeit des Programmcodes
14. Spezielle Abhandlung für kritische Parameter (nur durch Mitarbeiter Fluxron möglich)

Priorität 2

15. Austauschen von Küchenlayouts und Gerätepositionen

Priorität 3

16. Konzeptionelle Unterstützung für Internet- / Cloudanbindung
17. Konzeptionelle Unterstützung für das Weiterleiten von Fehlermeldungen and Fluxron
18. Konzeptionelle Unterstützung für das Erheben von Nutzungsprotokollen

1.6. Non Functional Requirements

1. Die Applikation soll 30 - 50 Fluxron Geräte problemlos verwalten können. In einzelnen Spezialfällen kann es aber bis zu 150 Geräte geben. Die App sollte damit ebenfalls umgehen können, allerdings sind leichte Einschränkungen kein Problem.

2. Die Applikation soll modular aufgebaut sein, so dass zukünftige Erweiterungen leicht realisiert werden können.
3. Der Code soll so dokumentiert werden, dass eine Weiterentwicklung der Applikation durch die Fluxron Solutions AG möglich ist.
4. Die Applikation soll auf Android Geräten die bis zu 2 Jahre alt sind lauffähig sein.
5. Sowohl Bluetooth Classic als auch Bluetooth 4.0 sollen von der Applikation unterstützt werden.
6. Die Applikation soll so einfach bedienbar sein, dass keine Schulung der Benutzer erforderlich ist.
7. Die Applikation soll auch ohne bestehende Internet-Verbindung vollständig funktionsfähig sein.

1.6.1. Anforderungen an Android-Version

Aufgrund der alten Produktgeneration muss die App auf einer Android-Version laufen, welche klassische Bluetoothverbindungen unterstützt. Zudem sollte die App auch auf einer neuen Version von Android mit Bluetooth 4.0 Low Energy (LE) betrieben werden können.

Bluetooth 4.0 unterstützt sowohl klassische Bluetoothverbindungen, als auch die neuen Low Energy Verbindungen. Die klassischen Verbindungen sind rückwärtskompatibel mit den Vorgängerversionen.[1]

Android unterstützt Bluetooth 4.0 (inklusive LE) ab der API-Version 4.3[2]. Dies ist somit die minimal nötige Version, um die Kompatibilität zu gewährleisten.

Mit der Auswahl der Version 4.3 (Android JellyBean) können somit mehr als 50% der momentan im Umlauf [3] befindlichen Geräte unterstützt werden.

1.7. Systemtest Spezifikation [1]

Zur Überprüfung der Applikationsfunktionalität werden Systemtests sowohl vom Entwicklerteam als, auch bei der Fluxron Solutions AG durchgeführt.

1.7.1. Systemtest Abdeckung

Mit folgender Tabelle wird gezeigt das alle Functional Requirement (FR), Non Functional Requirement (NFR) und Use Cases durch Systemtests abgedeckt sind.

UC	**FR**	**NFR**	**Systemtest**
A	3, 4		ST1: Küche initialisieren
A	5		ST2: Küche anpassen
A			ST3: Küche löschen
B	6, 11		ST4: Protokoll auslesen
B2	6, 8, 9		ST5: Geräteparameter ändern
B2	10		ST6: Geräteparameter ändern mit Validierung
C			ST7: Kritische Geräteparameter ändern
	1		ST8: Kennwortschutz
	1		ST9: Kennwortschutz mit falschem Kennwort
A	2		ST10: Suchlauf mit Gruppierung
	15		ST11: Austausch von Küchenlayouts
		1	ST12: Küche mit grosser Anzahl Geräte (50)
		1	ST13: Küche mit grosser Anzahl Geräte (150)
		5	ST14: Bluetooth Unterstützung
		7	ST15: Keine Netzwerkkonnektivität

Tabelle 1.8.: Systemtest Abdeckung

Nicht abgedeckte Anforderungen

- FR 16-17 (3. Priorität) fehlen in der Systemtest Abdeckungs Tabelle, da sie sich nur konzeptionell auf das Projekt auswirken und nicht getestet werden können.
- NFR 2-6 sind nicht durch Systemtests überprüfbar.

[1] Im Abschnitt C des Anhangs werden die Arbeitspakete detailliert aufgelistet.

1.7.2. Systemtest Durchführung

Die Systemtests wurden im Verlauf der Arbeit nach jedem Meilenstein (siehe 2.1) durchgeführt, um den Fortschritt zu kontrollieren. Am 25.11.2015 wurde die Applikation bei Fluxron in Amriswil an voll funktionsfähigen Geräten getestet. Dabei wurden alle Systemtests erfüllt. Die Systemtests ST12 und ST13 konnten nur teilweise getestet werden, da keine Möglichkeit bestand, die Systemtests in einer echten Grossküche durchzuführen.

18.10	**01.11**	**10.11**	**22.11**	**25.11**[1]	**Systemtest**
x	x	✓	✓	✓	ST1: Küche initialisieren
x	x	✓	✓	✓	ST2: Küche anpassen
✓	✓	✓	✓	✓	ST3: Küche löschen
x	x	x	✓	✓	ST4: Protokoll auslesen
x	x	✓	✓	✓	ST5: Geräteparameter ändern
x	x	x	✓	✓	ST6: Geräteparameter ändern mit Validierung
x	x	✓	✓	✓	ST7: Kritische Geräteparameter ändern
x	x	x	x	✓	ST8: Kennwortschutz
x	x	x	x	✓	ST9: Kennwortschutz mit falschem Kennwort
x	✓	✓	✓	✓	ST10: Suchlauf mit Gruppierung
x	x	x	✓	✓	ST11: Austausch von Küchenlayouts
x	x	✓[2]	✓[2]	✓[2]	ST12: Küche mit grosser Anzahl Geräte (50)
x	x	✓[2]	✓[2]	✓[2]	ST13: Küche mit grosser Anzahl Geräte (150)
✓	✓	✓	✓	✓	ST14: Bluetooth Unterstützung
✓	✓	✓	✓	✓	ST15: Keine Netzwerkkonnektivität

Tabelle 1.9.: Systemtest Durchführungen

[1] Testtag bei Fluxron in Amriswil

[2] Getestet mit simulierten Geräten ohne echte Bluetooth Verbindung. Es bestand keine Möglichkeit in einer echten Küche mit 50-150 Geräten Tests durchzuführen.

2. Projektmanagement

2.1. Meilensteine

MS	Name	Resultate	Datum
MS1	Kickoff	Aufgabenstellung	09.09.2015
MS2	Ende Elaboration	Infrastruktur, Ausgangslage, Anforderungsspezifikation, Projektplan, Risikomanagement, Use Cases, Testspezifikation, Domainanalyse, Arbeitspakete	29.09.2015
MS3	Architekturprototyp	Prototyp mit Grundfunktionalität der Applikation	13.10.2015
MS4	Zwischenpräsentation	Kunde und Experte informiert	27.10.2015
MS5	Feature 75%	75% aller Features erarbeitet	10.11.2015
MS6	Produkt fertig	Alle geforderten Funktionen der Applikation sind implementiert.	22.11.2015
MS7	Ende Testphase	Testprotokolle sind vorhanden	06.12.2015
MS7	Abgabe Abstract	Das Abstract wurde an den Betreuer abgegeben.	10.12.2015
MS8	Abgabe Bericht	Der Bericht wurde an den Betreuer abgegeben.	18.12.2015

Tabelle 2.1.: Meilensteine

2.2. Projektplan [1]

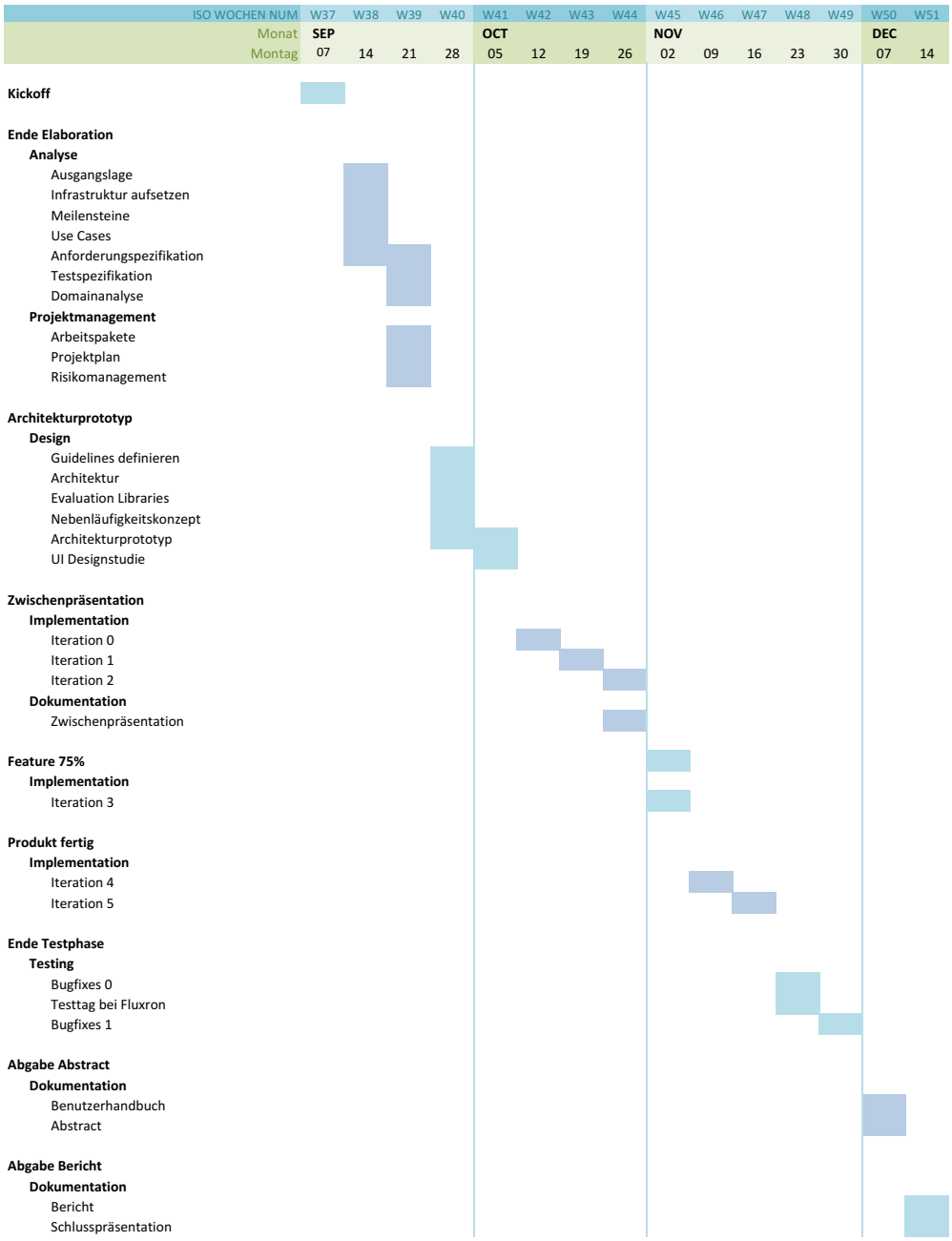


Abbildung 2.1.: Projektplan

[1] Im Abschnitt D des Anhangs werden die Arbeitspakete detailliert aufgeführt.

2.3. Risikomanagement

R1: Erwartungen des Kunden nicht erfüllt	
Beschreibung	Die Applikation erfüllt die funktionalen oder gestalterischen Erwartungen des Kunden nicht.
Massnahme	Es wird mit dem Kunden eine wöchentliche Besprechungen per Skype durchgeführt. Dabei wird der aktuelle Stand der Arbeit gezeigt.
Vorgehen beim Eintreffen	Applikation gemäss den Wünschen des Kunden anpassen.

Tabelle 2.2.: Risiko - Erwartungen des Kunden nicht erfüllt

R2: Performance reicht nicht für das Verwalten von 100 Geräten	
Beschreibung	Die Performance der Applikation reicht nicht aus um 100 Fluxron Geräte pro Küche zu verwalten beziehungsweise grafisch darzustellen.
Massnahme	Bei der Entwicklung werden Performance Tests mit bis zu 100 simulierten Fluxron Geräten durchgeführt.
Vorgehen beim Eintreffen	Die maximale Zahl von Geräten pro Küche reduzieren.

Tabelle 2.3.: Risiko - Performance reicht nicht

R3: Bluetooth Classic und 4.0 nicht in der gleichen Applikation	
Beschreibung	Es ist nicht möglich Unterstützung für Bluetooth Classic und 4.0 in der gleichen Applikation anzubieten.
Massnahme	In der Elaboration Phase wird die Bluetooth Unterstützung von Android recherchiert. Es wird nach Möglichkeit eine Android Version gewählt die beide Bluetooth Standards unterstützt.
Vorgehen beim Eintreffen	Es wird nur Unterstützung für Bluetooth Classic angeboten.

Tabelle 2.4.: Risiko - Bluetooth Classic & 4.0 Unterstützung

3. Konzeption und Design

3.1. Code Style Guidelines

Um die Zusammenarbeit mit mehreren Personen zu regeln, verwenden wir den folgende Code Style Guide:
`https://source.android.com/source/code-style.html#java-language-rules`.

Zur Vereinfachung einer späteren Veröffentlichung achten wir von Anfang an auf die Android Launch Checkliste:
`http://developer.android.com/distribute/tools/launch-checklist.html`.

3.1.1. Einschränkungen

Die folgenden Empfehlungen aus dem Code Style Guide werden wir nicht übernehmen:

- **Jedes File hat zuoberst ein Copyright Statement.**
  Der gesamte Code erhält eine Lizenz welche zentral abgelegt wird. Bei jedem File ein solches Statement einzufügen ist unnötig und wäre redundant.
- **Statische Variablen beginnen mit „s“. Nicht-öffentliche, nicht-statische Variablen beginne mit enquotem. Alle andern werden klein geschrieben.**
  Diese Konvention ist unnötig, da man diese Informationen problemlos auch dem Code entnehmen kann. Moderne IDEs wie Android Studio unterstützen dabei zusätzlich durch farbliches Hervorheben. In diesem Projekt werden alle Variablen, ausser Konstanten, klein geschrieben. Konstanten werden vollständig aus Grossbuchstaben zusammengesetzt.

3.2. UX Guidelines

Bei der Entwicklung des User Interfaces nutzen wir den Material Design Style Guide: `https://developer.android.com/design/index.html`. Dieser Guide ist eigentlich für Android 5.0 ausgelegt, unsere Applikation muss jedoch von Android 4.3 an lauffähig sein. Daher müssen einige Einschränkungen, z.B. bei der Verwendung von Animationen, in Kauf genommen werden.

3.3. Architektur

3.3.1. Rahmenbedingungen

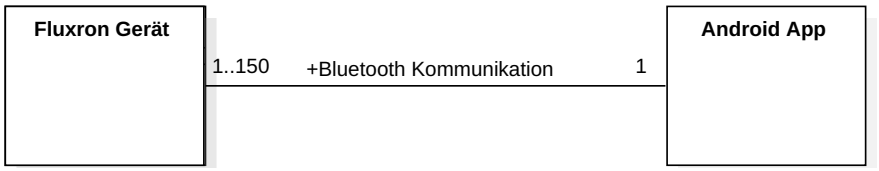


Abbildung 3.1.: Blackbox Darstellung

Die Applikation kommuniziert mit bis zu 150 Fluxron Geräten via Bluetooth. Aufgrund der Natur von Bluetooth Geräten ist mit einer stark asynchronen Kommunikation zu rechnen. Die Architektur der Fluxron Geräte ist bereits vorgegeben. Die Kommunikation zwischen den Geräten und der Applikation basiert auf CANopen.

Es soll eine Weiterentwicklung der Applikation durch die Fluxron Solutions AG möglich sein.

Die konzeptionellen Anforderungen der FR15-18 sowie NFR2, 3 sollen durch die Architektur unterstützt werden.

3.3.2. Variante A: Schichtenarchitektur

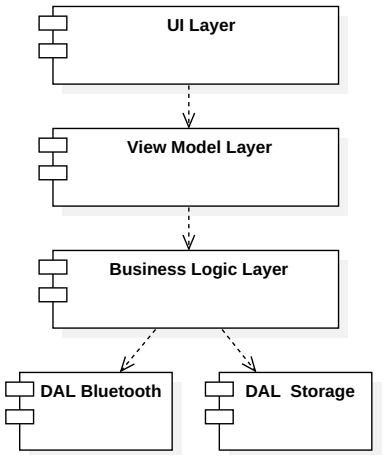


Abbildung 3.2.: Layer Architektur

Die Architektur wird in mehrere Schichten unterteilt. Dabei wird die Kopplung in die Gegenrichtung durch Interfaces und Observer aufgelöst.

3.3.3. Variante B: MVC Architektur

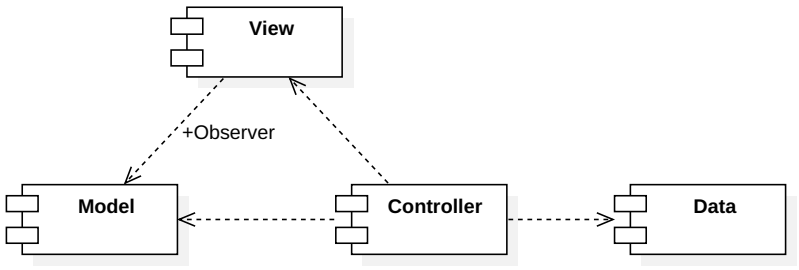


Abbildung 3.3.: MVC Architektur

Weil die Applikation stark durch das User Interface definiert ist, liegt eine Umsetzung mit dem MVC Pattern nahe.

3.3.4. Variante C: Event Bus Architektur

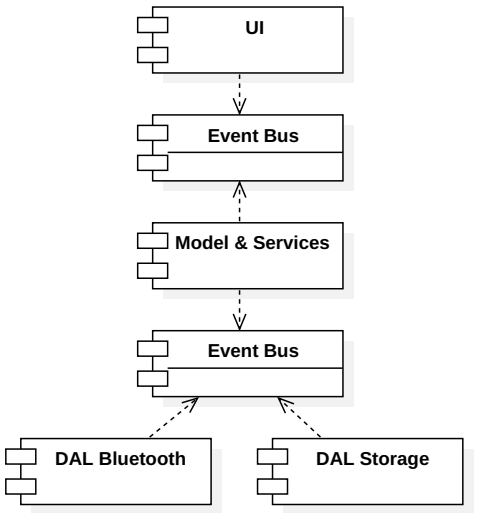


Abbildung 3.4.: Event Bus Architektur

Mit einem Event Bus zwischen UI und Model ist eine stärkere Entkopplung möglich[4]. Der Event Bus zwischen Model und Data Access Layer (DAL) ermöglicht mehrere Module wie Bluetooth und Storage.

3.3.5. Bewertung der Architekturmöglichkeiten [5]

Variante A: Schichtenarchitektur	
Vorteile	Nachteile
1. Einfach verständliche, klassische Architektur 2. Separation of Concerns, Low Coupling, High Cohesion	1. UI Layer muss ganze Objekthierarchien beobachten 2. Kein einheitliches Parallelitätskonzept
Variante B: MVC Architektur [6, 146 ff.]	
Vorteile	Nachteile
1. Gute Entkopplung der View von der Logik 2. Reduktion von Navigationslogik	1. Model View Controller (MVC) nur verwendbar in Kombination mit weiteren Patterns 2. Komplexe Observerstruktur zwischen View und Model 3. Kein einheitliches Parallelitätskonzept
Variante C: Event Bus Architektur	
Vorteile	Nachteile
1. Parallelität kann zentral von Bus synchronisiert werden 2. Einfachere und besser gekapselte Komponenten 3. Flache Observerstruktur im User Interface 4. Einfaches Hinzufügen von zusätzlichen Komponenten	1. Aufgrund flacher Hierarchie ist es leicht möglich Separation of Concerns zu verletzen. 2. Komplexere Interaktionslogik (Beispiel Message Chains)

Tabelle 3.1.: Bewertung Architekturmöglichkeiten

Fazit

Wir entscheiden uns für Variante C, denn sie ermöglicht uns die gewünschte konzeptionelle Erweiterbarkeit in Hinblick auf die FR und NFR, welche in Zukunft noch umgesetzt werden sollen. Zudem ermöglicht diese Architektur die Komplexität der Observer stark zu vereinfachen. Die stark asynchrone Kommunikation wird ideal unterstützt.

3.4. Nebenläufigkeitskonzept

Das Nebenläufigkeitskonzept bezieht sich auf die Architekturvariante C „Event Bus“.

3.4.1. Sicherstellen der Asynchronität

Um sicherzustellen, dass alle Ereignisse asynchron verarbeitet werden, muss der Event Bus so konfiguriert werden, dass er die Meldungsverarbeitung in einem eigenen Thread durchführt. Dies gilt auch für Meldungen die im UI-Thread erzeugt werden. So ist sichergestellt, dass der UI-Thread nicht blockiert.

3.4.2. Synchronisierung mit UI-Thread

Zugriffe auf das User Interface dürfen nur vom UI-Thread durchgeführt werden. Dazu müssen die Subscriber dem Event Bus den gewünschten Ausführungsthread angeben können.

3.4.3. Thread-Safety der Event Bus Subscriber

Alle Subscribers müssen sicherstellen (mit Ausnahme UI-Komponenten), dass sie von mehreren Threads aufgerufen werden können. Dazu werden klassische Java Monitors verwendet. Die Subscriber folgen dem „Run-to-completion“ Prinzip.

3.4.4. Restart von Activities

Android kann zur Speicheroptimierung laufende Applikationen (im Hintergrund) ganz oder teilweise beenden. Öffnet der Benutzer die Applikation erneut, muss sie ihren Zustand wiederherstellen. Dies erfordert eine Überprüfung, ob die Instanzen im Application Context noch vorhanden sind. Gegebenenfalls müssen die Event Busse neu gestartet werden.

3.5. Evaluation von Libraries

Zur Unterstützung der gewählten Architektur sind Libraries zu bevorzugen. In diesem Kapitel sind die Evaluationsergebnisse beschrieben.

3.5.1. Event Bus Library

Es stehen folgende zwei Libraries zur Auswahl:

1. Otto - An event bus by Square
2. Greenrobot EventBus

Diese sollen nun nachfolgend nach den, durch die Architektur vorgegebenen Kriterien bewertet werden.

Kriterium	**1: Otto**	**2: GreenRobot**
Instanzierung von mehreren Buses	x	x
Ausführung in eigenem Thread	(x)	x
Ausführung in UI-Thread	x	x
Ausführungsthread pro Subscriber wählbar		x
Neustart des Bus	x	x
Caching der letzten Events		x
Asynchrone Events		x
Eventproduzenten	x	

Tabelle 3.2.: Bewertung Event Bus Libraries

Bewertung und Fazit

Mit den nicht vorhandenen Kriterien „Ausführungsthread pro Subscriber wählbar“ und „Asynchrone Event“ fehlen der Library „Otto“ zwei unverzichtbare Features. Da das einzige Feature, welches von GreenRobot nicht unterstützt wird, die Eventproduzenten sind (diese können durch Request/Response einfach ersetzt werden), kann der GreenRobot EventBus eingesetzt werden.

GreenRobot EventBus ist unter der Apache License [7] Version 2.0 lizenziert. Diese erfordert lediglich eine Kopie der Lizenz mit der gelieferten Software. Da keine Modifikationen an der eigentlichen Library gemacht wird, ist dies die einzige Einschränkung.

3.5.2. Lokaler Speicher für die Küchendaten und Settings

Zur lokalen Speicherung der Daten soll eine Datenbankbibliothek eingesetzt werden. Da die Küchendaten immer als ganze Küche geladen werden, soll eine dokumentbasierte Datenbank zum Einsatz kommen. Momentan sind die folgenden Libararies aktuell und sind daher in Betracht zu ziehen:

1. Oracle BerkeleyDB
2. Couchbase Lite
3. UnQLite

In der folgenden Tabelle sind die wichtigsten Kriterien mit einer Bewertung aufgelistet.

Kriterium	**1: BerkeleyDB**	**2: Couchbase**	**2: UnQLite**
Android API 18 Support	x	x	(x)
Komerzielle Nutzung		x	x
Queries	x	x	x
Pagination	x	x	x
Replication	x	x	
Dokumentationsqualität	+	+	-

Tabelle 3.3.: Bewertung dokumentbasierte Datenbank für Android

Bewertung und Fazit

Da BerkeleyDB von Oracle für kommerzielle Nutzungszwecke eine, von Oracle genehmigte Einzellizenz benötigt, kann diese Library nicht verwendet werden. Die unzureichende Dokumentation und fehlende Replikationsfeatures machen UnQLite ebenfalls zu einem ungeeignetem Kandidaten.

Damit fällt die Wahl auf Couchbase Lite. Diese Bibliothek bietet auch im Hinblick auf die optionalen NFR eine solide Basis.

4. UI-/UX-Konzept

4.1. Farben und Formen

4.1.1. Farbgebung

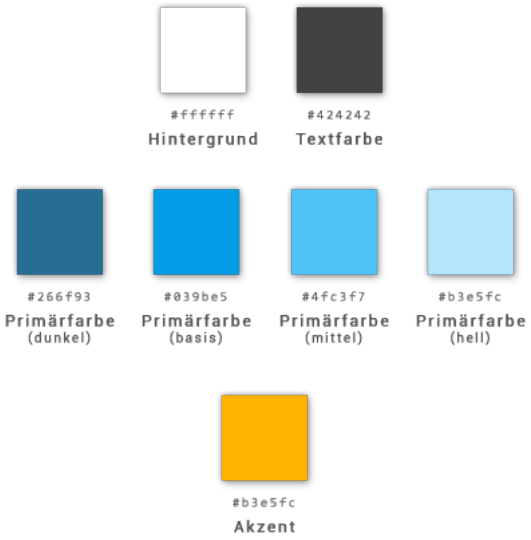


Abbildung 4.1.: Grundlegendes Farbschema der Anwendung

Die Farbgebung inspiriert sich am Logo der Firma Fluxron. Ein weisser Hintergrund, mit dunkelgrauer Schrift. Zudem wird ein einfaches aber aussagekräftiges Schema aus Komplementärfarben zur Hervorhebung spezieller Elemente angewendet.

4.1.2. Farben zur Statusanzeige

Abbildung 4.2.: Farben zur Statusanzeige

Zur Anzeige von Statusinformationen wird ein klassisches Ampel-Farbschema angewendet. Dieses soll durch die Nutzung von Symbolen unterstützt werden um die Accessibility zu verbessern. Diese Farben sollten nur für kleine Flächen genutzt werden.

4.1.3. Symbole

Um die Usability zu erhöhen werden neben den Farben auch Symbole zur besseren Kommunikation verwendet. Abbildung 4.3 zeigt eine Auflistung dieser Symbole.

Abbildung 4.3.: Symbole des User-Interface

4.2. Navigationskonzept

Abbildung 4.4 zeigt das Navigationskonzept für die Mobilapplikation. Zur Navigation verwendet der Benutzer Menüschaltflächen, Listen sowie den Hardwarebutton zur Rückwärtsnavigation („Back-Button“).

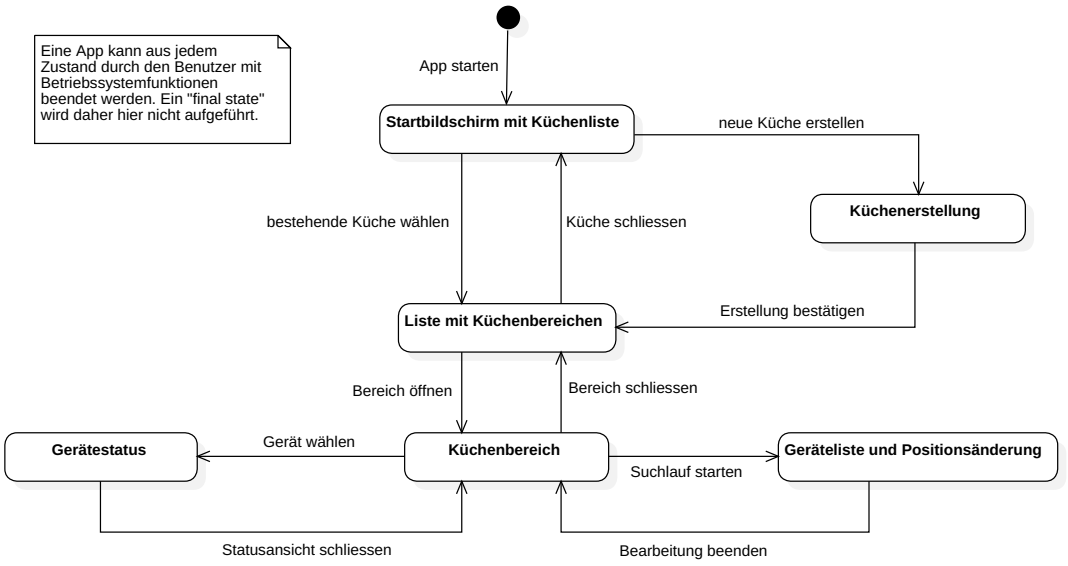


Abbildung 4.4.: Farben zur Statusanzeige

4.3. Mockups

4.3.1. Startbildschirm

Der Startbildschirm enthält ein Suchfeld zur einfachen Suche einer Küche nach dem Namen. Darunter werden die Küchen mit ihrem Namen, der Beschreibung und mit der Anzahl Geräte aufgelistet.

Der Benutzer kann nun eine Küche durch Antippen öffnen, oder mittels dem Floating Action Button (FAB) eine neue Küche hinzufügen.

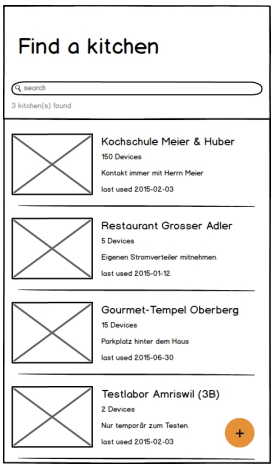


Abbildung 4.5.: Mockup Startbildschirm

4.3.2. Küchenerstellung

Bei der Küchenerstellung gibt der Benutzer einen Namen für die Küche an. Zudem kann der Benutzer, wenn er möchte, noch eine Beschreibung für diese Küche eingeben.

Zusätzlich sollte der Benutzer auch noch ein Foto von der Küche machen. Damit kann er die Küche später einfacher wiedererkennen. Küchenname und Bild müssen zwingend erfasst werden, die Beschreibung ist nicht erforderlich.

Durch Antippen des „Create“-Buttons wird die Küche erfasst und der Benutzer auf die Küchenansicht umgeleitet.

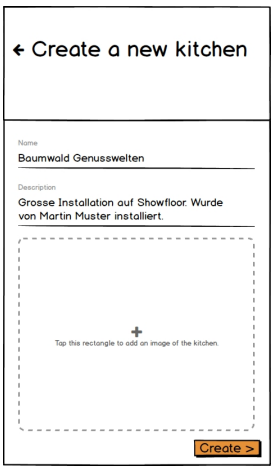


Abbildung 4.6.: Mockup Küchenerstellung

4.3.3. Küchenbereich mit aktivem Suchlauf

In der Küchenansicht sieht der Benutzer das Foto für den aktuell gewählten Küchenbereich. Darauf kann er die Geräte aus der Geräteliste platzieren. Die Geräte werden durch einen Kreis, welcher den Status des Gerätes anzeigt, symbolisiert.

Durch Antippen eines Gerätes kann dieses nach Rückfrage gelöscht werden.

Die Geräteliste enthält alle Geräte, welche beim Suchlauf gefunden wurden, oder bereits in der Küche erfasst wurden.

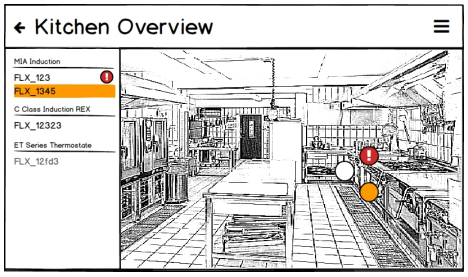


Abbildung 4.7.: Mockup Küchenansicht

4.3.4. Gerätestatus

Die Statusansicht für ein Gerät besteht aus Tabs, zwischen denen der Benutzer mittels Swiping wechseln kann. Für jeden Gerätetyp braucht es hier eine eigene Ansicht, da für jede Art von Gerät andere Parameter und Werte relevant sind.

- Statusübersicht mit Sensorwerten
- Nutzungsstatistik des Geräts
- Fehlerhistorie mit letzten 10 Fehlercodes
- Einstellungsseite mit Konfigurationsparametern

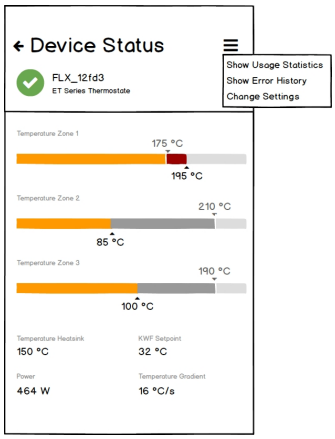


Abbildung 4.8.: Mockup Gerätestatus

4.3.5. Geräteparameter

Die Parameteransicht zeigt eine Liste aller veränderbaren Konfigurationseinstellungen an. Der Benutzer kann die Parameter ändern oder zurücksetzen.

Falls ein Parameter nicht geschrieben werden kann, wird eine Fehlermeldung ausgegeben.

Diese Ansicht ist ebenfalls für jeden Gerätetypen unterschiedlich, so haben z.B. Thermostaten mehrere Konfigurationsprofile, welche alle in dieser Ansicht bearbeitet werden können.

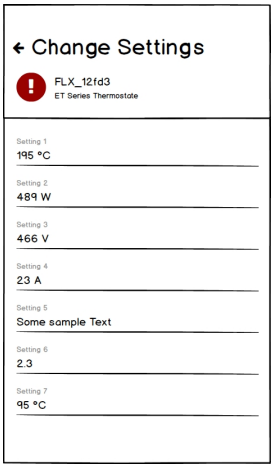


Abbildung 4.9.: Mockup Geräteparameter

4.3.6. Nutzungsstatistik

Über die Nutzungsstatistik kann der Benutzer auslesen, wie lange die Werte der Sensoren in den einzelnen Temperaturbereichen lagen. Dies kann zum Beispiel Auskunft über eine fehlerhafte Installation (nicht genügend Wärmeabfuhr am Kühlkörper) Aufschluss geben.

Auch die Statistikseite kann von Gerätetyp zu Gerätetyp variieren.

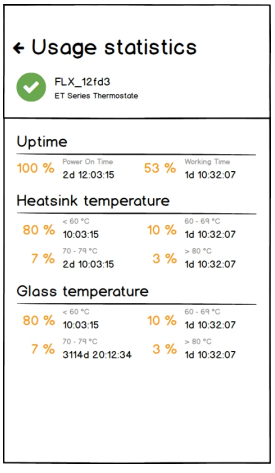


Abbildung 4.10.: Mockup Nutzungsstatistik

4.3.7. Fehlerhistorie

In der Fehlerhistorie werden dem Benutzer die letzten zehn Fehlercodes des Gerätes angezeigt. Diese Bestehen aus einem Fehlercode und einem beschreibenden Text. Zudem wird der Laufzeitzähler des Gerätes zum Fehlerzeitpunkt ausgegeben. Das heisst, für jeden Fehler kann nachgesehen werden, in welcher Laufzeitstunde (ab Installationszeitpunkt) der Fehler aufgetreten ist.

Einzelne Gerätetypen wie z.B. Thermostaten haben weniger oder mehr Fehlercodes in der Historie.

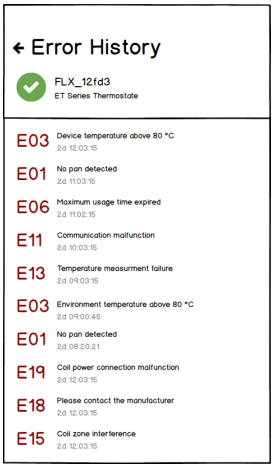


Abbildung 4.11.: Mockup Fehlerhistorie

5. Ergebnisse

5.1. Software

5.1.1. Umfang

Im Rahmen des Projektes wurde eine Android-Applikation entwickelt. Diese läuft ab der Androidversion 4.3 (API Level 18). Mit der Applikation können Servicetechniker Küchenaufbauten protokollieren und die Parametrisierung der Geräte auslesen und bearbeiten.

Die App wurde von der Firma Fluxron Solutions AG übernommen und wird dort weiterentwickelt und in den App Store publiziert.

5.1.2. Package-Struktur

Die Software basiert auf einem Schichtenmodell, bei welchem die Schichten mittels einem Event Bus verbunden werden. Zudem gibt es einen Applikationskontext, Pakete mit Meldungstypen und ein Paket mit den Datenobjekten.

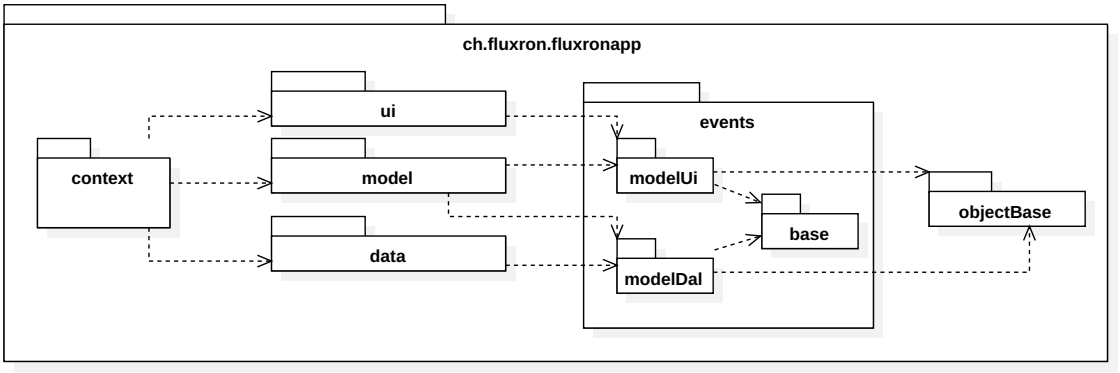


Abbildung 5.1.: Packagediagramm der App

ch.fluxron.fluxronapp.context

Dieses Paket enthält den Applikationskontext. Dieser wird beim Starten der Applikation initialisiert und beinhaltet sowohl den Event Bus für die Kommunikation zwischen Model und User Interface, als auch den Bus zwischen Data und Model. Die Referenzen auf den jeweiligen Event Bus werden über Dependency Injection an die, ebenfalls im Kontext initialisierten, Listenerklassen aus Model und Data weitergegeben.

ch.fluxron.fluxronapp.ui

Beinhaltet alle UI-Klassen. Dies sind Activities, Fragments, Custom Views, Animationen und Listenadapter. Diese Klassen haben alle direkt mit dem User Interface und der Anzeigelogik zu tun.

ch.fluxron.fluxronapp.model

In diesem Paket sind die Logikklassen des Businessmodells gebündelt. Sie alle haben gemeinsam, dass sie Meldungen von der Benutzeroberfläche verarbeiten und an die Datenschicht weiterleiten (oder umgekehrt).

ch.fluxron.fluxronapp.data

Dieses Paket besteht aus zwei Hauptbereichen: Die lokale Datenbankanbindung mittels Couchbase Lite und die Bluetooth-Schnittstelle. Beide Bereiche erhalten Meldungen von der Logikschicht und lösen selbst Meldungen aus (z.B. ein Gerät wurde gefunden).

ch.fluxron.fluxronapp.events

Die gesamte Kommunikation zwischen den Layern basiert auf Event- und Meldungsklassen. Diese sind in diesem Paket in drei Unterpakete gegliedert.

Das Paket **modelUi** enthält Nachrichten, welche zwischen Benutzeroberfläche und Logik ausgetauscht werden.

Im Package **modelDal** befinden sich Meldungstypen, die für die Kommunikation zwischen Logik, Datenbank und Bluetooth-Schnittstelle versendet werden.

Beide Meldungsgruppen basieren auf den Klassen aus **base**. Diese implementieren grundlegende Mechanismen wie z.B. Request-Reply Pattern oder das synchrone Warten auf Antworten nach einem Request.

ch.fluxron.fluxronapp.objectBase

Für den Nachrichtenaustausch und die Speicherung der Daten werden noch Datenobjekte benötigt. Diese sind häufig Bestandteil einer Meldung und werden somit von vielen Meldungsklassen referenziert.

5.1.3. Laufzeitstruktur

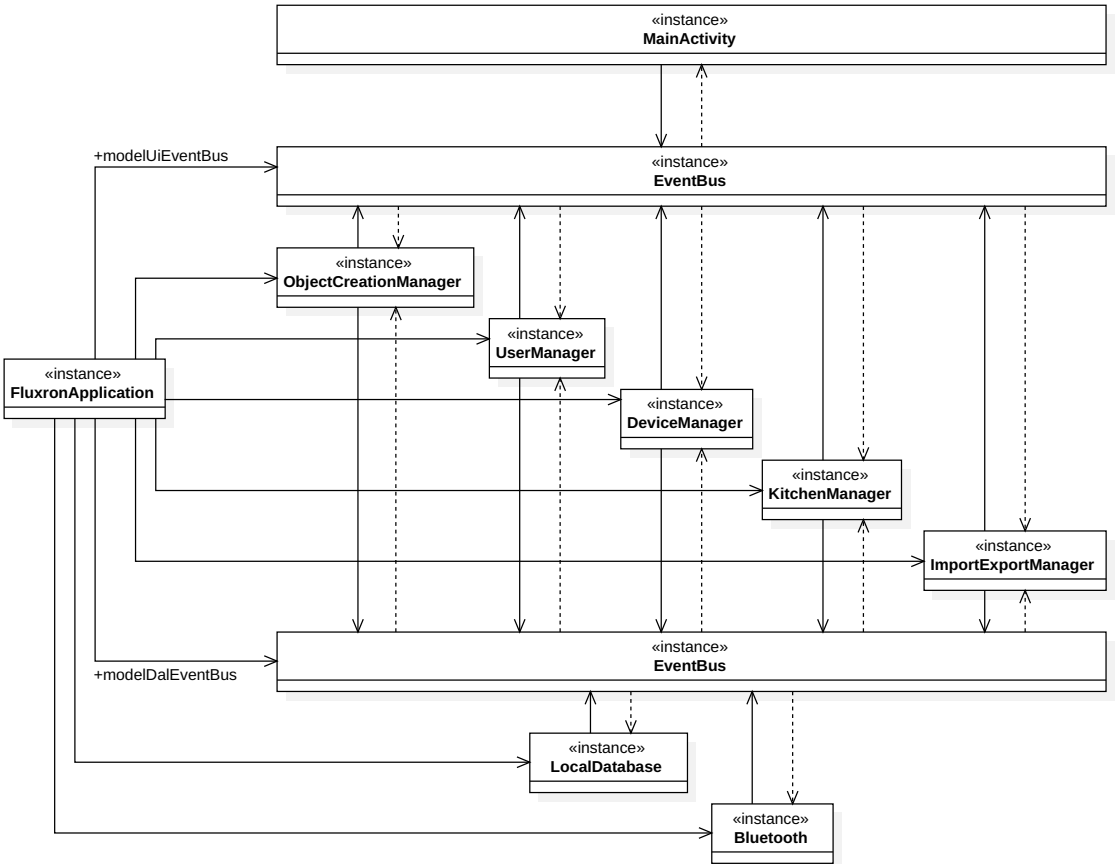


Abbildung 5.2.: Laufzeitstruktur der App

Beim Start der Applikation wird der Kontext (FluxronApplication) durch das Betriebssystem erzeugt. Dieser erstellt dann die beiden Event Bus Objekte und erzeugt auch die Instanzen der Manager- und Datenklassen. Dies ist notwendig, da sich die Layer gegenseitig nicht kennen und nur über einen Event Bus miteinander kommunizieren können. Die Instanziierung aller Manager muss direkt zu Beginn der Applikation gemacht werden, damit sich die Instanzen beim Event Bus registrieren können.

Jede Managerklasse hat eine Assoziation mit dem Event Bus (Aufruf des Meldungsversandes) und wird durch diesen aufgrund der Subscription referenziert.

5.1.4. Nachrichtenfluss

Nachrichten werden auf mehrere Arten genutzt, um die Verbindung zwischen den Layern herzustellen:

- Versenden von Befehlen (z.B. SaveKitchenCommand)
- Kontextlose Ereignismeldungen (z.B. DeviceFound)
- Antworten auf Befehle (z.B. ResponseOK)

Nachfolgend ist der Nachrichtenverlauf am Beispiel „Ändern im Fehlerfall“ abgebildet.

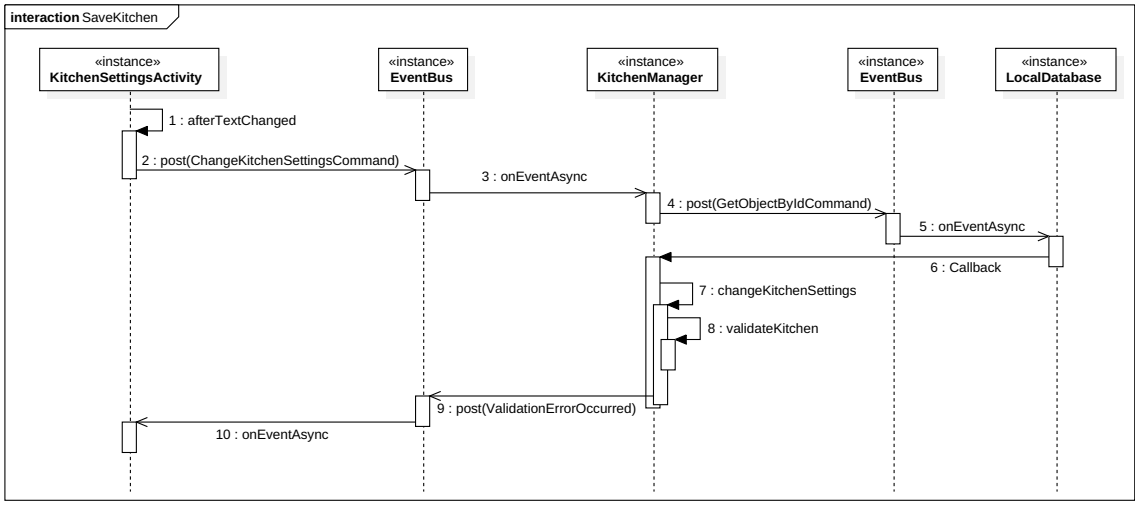


Abbildung 5.3.: Meldungsverlauf „Ändern von Kücheneinstellungen mit Validierungsfehlern“

1. Benutzeroberfläche registriert eine Textänderung
2. Wertänderung Name wird als Befehl über den Event Bus versendet
3. Bus verteilt die Meldung an alle Listener
4. Ladebefehl für die Küche anhand Id
5. LocalDatabase empfängt Ladebefehl
6. Callback wird aufgerufen, sobald das Objekt geladen ist
7. Einstellungen des geladenen Objekts werden angepasst
8. Validierung der Werte
9. Aufgetretene Fehlermeldungen werden als Event über den Bus versendet
10. Meldung trifft bei Benutzeroberfläche ein, Fehler wird dargestellt

5.1.5. Codestatistik

Zur Überprüfung der Codequalität wurde neben Codereviews auch Codestatistiken ausgewertet. Nachfolgend ist die Statistik für das Projektende aufgeführt. Alle gemessenen Werte befinden sich im erwarteten und grünen Bereich. Im Schnitt zeigen diese Werte eine gute Wartbarkeit und Qualität des entwickelten Programms.

Bei einzelnen Klassen sind Abweichungen vom Durchschnitt festzustellen. Diese lassen sich aber alle begründen. So sind zum Beispiel die Werte für die „Lack of Cohesion of Methods“ der Plain Old Java Objects sehr hoch, denn diese bestehen nur aus Gettern und Settern.

Class Metrics

Metrik	**Wert**
(Avg) Coupling Between Objects	7.30
(Avg) Depth of Inheritance	2.35
(Avg) Lack of Cohesion of Methods	2.04
(Avg) Weighted Method Complexity	9.34
Number of Product Classes	217.00

Tabelle 5.1.: Class Metrics für ch.fluxron.fluxronapp

Hier stechen zwei Metriken besonders hervor. Die hohe Klassenzahl ist durch die Meldungsklassen erklärbar. Da jede Aktion mit einem Command und mindestens einer Response umgesetzt werden muss, ergibt dies eine hohe Klassenzahl. Dies ist allerdings nicht problematisch, sind die Meldungsklassen doch meist nicht mehr als simple Klassen mit ein bis zwei Variablen ohne Logik.

Die hohe Weighted Method Complexity erklärt sich mit der Verwendung des Event Bus. Jeder Meldungstyp muss mit einer Methode abonniert werden. Die Klassen, welche Meldungen verarbeiten erhalten somit viele Methoden. Gerade im Model macht es keinen Sinn, für jede Meldungsklasse eine eigene Abonnentenklasse zu definieren. Diese Manager haben daher eine hohe Metrik, sind aber nicht problematisch.

Complexity Metrics

Metrik	Wert
(Avg) Method Cyclomatic Complexity	1.56
(Avg) Method Lines Of Code	11.69
(Avg) Comment to Code Ratio	38.56
(Avg) Javadoc Lines per Method	4.23

Tabelle 5.2.: Complexity Metrics für ch.fluxron.fluxronapp

Hier sind alle Metriken in sehr guten Bereichen. Die Metriken bescheinigen der Anwendung eine hohe Wartbarkeit und zeigen wunderbar auf, wie ein Event Bus die Architektur und Komplexität der Applikation vereinfachen kann.

Dependency Metrics

Metrik	Wert
(Avg) Number of Cyclic Dependencies	0.00
(Avg) Number of Class Dependencies	4.33
(Avg) Number of Package Dependencies	0.96

Tabelle 5.3.: Dependency Metrics für ch.fluxron.fluxronapp

Hier fällt die Anzahl Klassenabhängigkeiten auf. Diese ist aber mit 4 immer noch auf sehr gutem Niveau.

5.2. Zielerreichung

Im Rahmen dieser Arbeit wurde eine Android Applikation entwickelt, welche dem Stand der Technik entspricht und auf allen gängigen Android Smartphones der letzten 2 Jahre lauffähig ist. [1] Alle beim Kick-Off Meeting ausgearbeiteten Anforderungen des Kunden (siehe 1.5) wurden erfolgreich umgesetzt und getestet.[2]

Kurz zusammengefasst hat die Applikation die folgenden Features:

- Ein Servicetechniker kann eine neue Küche erfassen und diese mit Beschreibung und Bild versehen. In der erfassten Küche können dann Fotos der verschiedenen Küchenbereiche gespeichert werden.
- Durch ein Bluetooth Discovery Vorgang werden alle vorhandenen, aktiven Fluxron Küchengeräte erkannt. Diese können frei auf den Fotos der Küchenbereiche platziert werden.
- Nachdem Geräte erfasst wurden ist auf den Fotos der Küchenbereiche jederzeit der Betriebsstatus ersichtlich.
- Detailinformationen wie Temperatur, Betriebsmodus, Fehlermeldung und Ähnliches werden sichtbar, wenn man auf das Icon eines Geräts tippt. Auch Änderungen an der Konfiguration des Geräts sind möglich.
- Bereits erfasste Küchen können zwischen den Servicetechnikern einer Firma durch eine Export/Import Funktion per Mail ausgetauscht werden.
- Weiter sind auch alle Create Read Update Delete (CRUD)-Operationen unterstützt.

Da eine Weiterentwicklung durch die Fluxron vorgesehen ist, wurde bei der Programmierung speziellen Wert darauf gelegt, dass die Applikation gut dokumentiert und leicht erweiterbar ist. Der JavaDoc Abdeckungsgrad liegt bei nahezu 100%.

[1] Android 4.3 und höher wird unterstützt.
[2] Anforderungen der Priorität 3 sind konzeptionell unterstützt.

5.3. Benutzerhandbuch

5.3.1. Installation

Um die Applikation zu installieren, muss die .apk Installationsdatei zuerst auf das Smartphone übertragen werden. Danach muss sichergestellt werden, dass in den Android Einstellungen unter Sicherheit die Option „Unbekannte Herkunft zulassen“ aktiviert wurde. Nun kann mit einem gängigen Dateiexplorer die Installationsdatei geöffnet werden um den Installationsvorgang zu beginnen.

5.3.2. Logininformationen hinterlegen

Um die Applikation verwenden zu können, müssen die Login-Informationen des Benutzers hinterlegt werden. Dazu wird direkt nach dem Start der Applikation das Zahnrad im oberen rechten Ecken berührt.

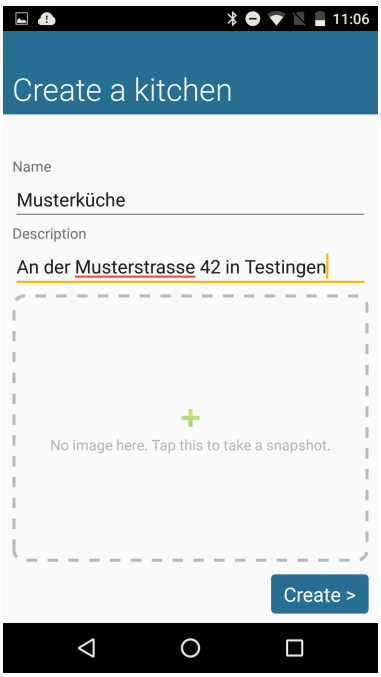


Abbildung 5.4.: Küche erstellen

Dies öffnet die Applikationseinstellungen, wo Nutzername und das Passwort eingegeben werden können.[3]

5.3.3. Küche erfassen

Um eine Küche zu erfassen, berührt man den Floating Action Button in der unteren rechten Ecke der Ansicht „Find a kitchen“. Danach wird der Name der Küche sowie eine optionale Beschreibung erfasst. Zum Schluss wird noch ein Foto der Küche erstellt, um eine schnelle Wiedererkennung zu ermöglichen.

Nun können für die neu erstellte Küche verschiedene Küchenbereiche erfasst werden. Dazu wird in der aktuellen Ansicht der Floating Action Button

[3]Zu Testzwecken kann der Benutzername: „user“ und das Passwort: „1234“ verwendet werden.

in der unteren rechten Ecke berührt. Dies kann so lange wiederholt werden, bis alle Bereiche der Küche erfasst wurden.

5.3.4. Suchlauf durchführen

Nachdem eine Küche mit mindestens einem Küchenbereich erfasst wurde, können Geräte auf dem Bereich platziert werden. Dazu berührt man zuerst das Bild eines Küchenbereichs. Um in diesem Bereich nun einen Suchlauf durchzuführen, muss das Bluetooth Symbol in der oberen rechten Ecke berührt werden. Dieses öffnet eine Liste, welche nun laufend mit den gefunden Geräte befüllt wird.

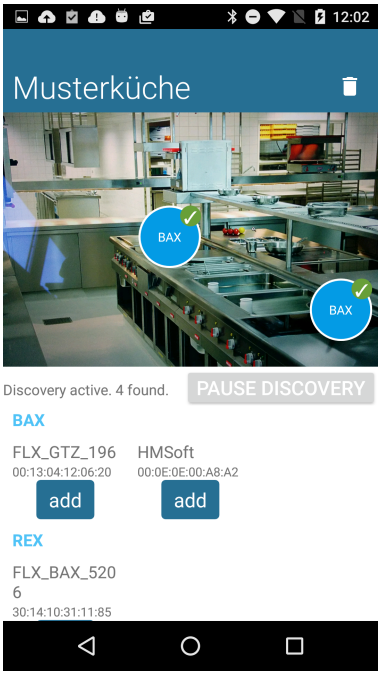


Abbildung 5.5.: Geräteerkennung

Geräte, mit denen noch keine Verbindung bestand, zeigen den Button „Pair“ an. Damit wird der Bonding-Vorgang gestartet, welcher kurzzeitig ein Pop-Up für eine Passworteingabe anzeigt. Bei Geräten, die mit dem Default Passwort „1234“ konfiguriert sind, muss nichts weiter gemacht werden, das Pop-Up verschwindet nach einigen Sekunden automatisch und der Button zeigt daraufhin „Add“ an.

Nach dem Bonding-Vorgang können Geräte auf dem Küchenbereich platziert werden. Dazu berührt man den „Add“ Button, worauf das Gerät in der Mitte der Ansicht erscheint. Das Gerät kann auf der Ansicht frei verschoben werden, so dass es mit der Einrichtung der Küche übereinstimmt.

Sobald alle Geräte erfolgreich platziert wurden, kann der „Discovery Modus“ mit dem Backbutton des Smartphones verlassen werden.

5.3.5. Gerätestatus abrufen

Um den Status eines Geräts abzurufen, muss man sich in der Küchenbereichsansicht befinden. Danach öffnet eine Berührung des gewünschten Geräts die Geräteansicht. Hier hat man die Wahl zwischen vier Tabs: Status, Usage, Errors und Config.

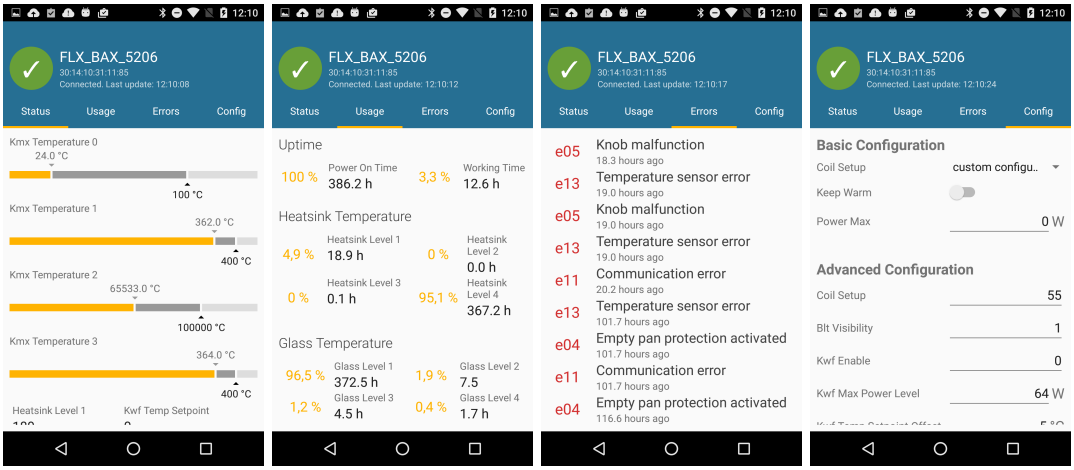


Abbildung 5.6.: Geräte-Detailansicht

- **Status:** In der Statusansicht wird die aktuelle Temperatur aller angeschlossenen Sensoren angezeigt.

- **Usage:** Unter Usage findet man die Uptime des Geräts sowie die Benutzungsdauer einiger der wichtigsten Komponenten.

- **Errors:** Der Tab Errors zeigt eine Liste der Fehler die auf dem Gerät aufgetreten sind.

- **Config:** Unter Config können Geräteinstellungen eingesehen und verändert werden. Config ist in eine Überschrift „Basic Configuration“ mit nur den nötigsten Einstellungen sowie „Advanced Config“ mit allen Einstellungen gegliedert.

5.3.6. Küche editieren

Ausgehend vom Screen „Find a kitchen“ wird eine Küche gewählt und auf das Zahnrad in der oberen rechten Ecke gedrückt. Dies öffnet die Kücheneinstellung, in welcher der

Name und die Beschreibung der Küche geändert werden können. Änderungen werden automatisch gespeichert.

5.3.7. Küche exportieren

Um eine Küche zu exportieren muss zuerst zu den Kücheneinstellungen navigiert werden (siehe 5.3.6). Dort kann via dem Button „Share via EMail“ der Export-Vorgang gestartet werden. Nachdem die Applikation den Export zusammengestellt hat, erscheint eine Auswahl von Applikationen mit denen die exportierte Küche versendet oder gespeichert werden kann. Wenn zum Beispiel Gmail gewählt wird, wird die exportierte Küche als .fluxron Datei angehängt.

5.3.8. Küche importieren

Um eine Küche zu importierten, benötigt man eine .fluxron Datei (siehe 5.3.7). Diese Datei kann auf ein Smartphone via File Transfer übertragen und dann mit einem herkömmlichen Datei Explorer geöffnet werden. Alternativ kann man sich Datei auch als EMail Anhang zusenden und direkt aus der Mail Applikation heraus öffnen.

5.3.9. Deinstallation

Die Deinstallation erfolgt durch ein langes Drücken des Applikationsicons. Das Icon kann dann auf den Mülleimer gezogen werden, der am oberen Bildschirmrand erscheint. Alternativ kann die Applikation auch in den Android-Einstellungen unter „Apps“ deinstalliert werden.

6. Ausblick

6.1. Weiterentwicklungsmöglichkeiten

6.1.1. Kommunikation mit weiteren Gerätetypen

In der aktuellen Version der App wurden nur die Gerätetypen C- und S-Class vollständig angebunden. Zukünftig sollten aber auch weitere Fluxron-Geräte angesprochen werden können.

Dies wird von unserer App bereits konzeptionell unterstützt, d.h. es müssen lediglich die Benutzeroberflächen für die entsprechenden Gerätetypen gezeichnet werden. Je nach Gerätetyp muss auch noch die Code-Generierung um die entsprechenden Parameterdateien erweitert werden. Die App unterscheidet die Geräte bereits an ihrem Herstellercode und kann daher um beliebige Gerätetypen erweitert werden.

6.1.2. Internet- / Cloudanbindung

Da die Anwendung typischerweise von Angestellten einer Servicefirma genutzt wird, ist die aktuelle Möglichkeit, Küchen via E-Mail auszutauschen nicht langfristig praktikabel. Daher wird eine Internetanbindung der App zusätzlichen Nutzen verleihen. Die Firmen könnten damit die erfassten Küchen an alle Mitarbeiter verteilen und müssten sich nicht mehr um eine Datensicherung der einzelnen Geräte kümmern. Zudem wäre die Verwaltung aller Küchen einer Servicefirma über ein Webportal denkbar.

Aber nicht nur für die Servicefirmen hätte eine solche internet- bzw. cloudbasierte Lösung Vorteile. Auch die Firma Fluxron könnte von dieser profitieren. Es könnten zusätzliche Einnahmen mit dem Vermieten des Cloudservers an die Servicefirmen erzielt werden.

Eine solche Anbindung ist mit der von uns gewählten Applikationsarchitektur problemlos möglich. Dabei kann einerseits die Datenbank (Couchbase Lite) mit einem Cloud-Backend verbunden werden. Dies ist von Couchbase bereits im Rahmen von Merge-Replikation unterstützt und ist damit mit geringem Aufwand möglich.

Mit etwas mehr Entwicklungsaufwand könnte aber auch eine komplett neue Komponente zur Internetanbindung an den Daten-EventBus angehängt werden. Diese würde dann die empfangenen Meldungen an ein Cloud-Backend weiterleiten und damit theoretisch sogar eine Echtzeitanbindung ermöglichen.

6.1.3. Erstellen eines Geräteabbilds

Ähnlich der momentanen Exportmöglichkeit für Küchen, wäre eine Erweiterung der App denkbar, welche es erlaubt, ein Abbild eines Gerätes zu erstellen. Dieses Geräteabbild könnte die Fehlerhistorie, Nutzungsstatistiken und alle Parameter (mit Werten) des Gerätes enthalten. Danach könnte das exportierte Abbild an die Herstellerfirma gesendet werden, um dort eine Fehlerdiagnose zu machen.

Neben dem Export eines Geräteabbildes wäre natürlich auch der Import eines Geräteabbildes von Nutzen. So könnten Mitarbeiter der Firma Fluxron Konfigurationen vor Ort vom Handy auf mehrere Geräte überspielen, oder die Konfiguration an eine Servicefirma weiterleiten, welche dann die Einstellungen auf die, beim Kunden installierten Geräte, übertragen kann.

6.1.4. Erhebung von Nutzungsprotokollen

Verbindet man die Cloudanbindung (6.1.2) und die Erstellung von Geräteabbildern (6.1.3), wäre auch eine Erhebung von Nutzungsprotokollen der Geräte möglich. Damit kann die Firma die Leistung der Geräte für den Kundenbedarf anpassen und optimieren.

Ausserdem könnten auch die Servicefirmen von einer solchen Erhebung profitieren. Sie können so einfacher Probleme diagnostizieren und Support für ihre Kunden anbieten. Häufig bauen die Servicefirmen die Fluxron-Geräte in eigene Küchenkombinationen ein, welche dann weiterverkauft werden. Mittels der Nutzungsstatistik kann so der

Marktbedarf besser eingeschätzt werden.

Ein weiterer Nutzen der Statistiken wäre es, Vorhersagen über mögliche nahende Probleme eines Gerätes machen zu können. So könnten die Geräte ausgetauscht oder repariert werden, bevor die Probleme im laufenden Betrieb auftreten.

Literaturverzeichnis

[1] "Bluetooth specification version 4.0," 2010. [Online]. Available: https://www.bluetooth.org/docman/handlers/downloaddoc.ashx?doc_id=229737

[2] "Bluetooth low energy | android developers." [Online]. Available: https://developer.android.com/guide/topics/connectivity/bluetooth-le.html

[3] "Dashboards | android developers." [Online]. Available: https://developer.android.com/about/dashboards/index.html

[4] "Event collaboration," 2006. [Online]. Available: http://martinfowler.com/eaaDev/EventCollaboration.html

[5] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design patterns: elements of reusable object-oriented software*. Pearson Education, 1994.

[6] H. Mössenböck, *Object-oriented programming in Oberon-2*. Springer Science & Business Media, 2012.

[7] "Apache license version 2.0," 2004. [Online]. Available: http://www.apache.org/licenses/LICENSE-2.0

[8] "Android tools project site," 2014. [Online]. Available: http://tools.android.com/download/studio/stable

Abkürzungsverzeichnis

FAB	Floating Action Button
IDE	Integrated Development Environment
HSR	Hochschule für Technik Rapperswil
UI	User Interface
LE	Low Energy
FR	Functional Requirement
UC	Use Case
NFR	Non Functional Requirement
DAL	Data Access Layer
UI	User Interface
MVC	Model View Controller
VPS	Virtual Private Server
CRUD	Create Read Update Delete

Abbildungsverzeichnis

Tabellenverzeichnis

Anhang

A. Infrastruktur

A.1. Entwicklungsumgebung

Als Integrated Development Environment (IDE) wurde für dieses Projekt Android Studio 1.4 eingesetzt. Android Studio basiert auf IntelliJ und hat Ende 2014 die „Eclipse Android Development Tools“ als Standard IDE für Android Projekte abgelöst.[8]

Für die Versionsverwaltung wurde Git mit einem Projekt Repository der Hochschule für Technik Rapperswil (HSR) (`git.hsr.ch`) verwendet. Lokal wurde der grafische Git-Client SourceTree benutzt.

Um Continuous Integration zu ermöglichen haben wir zusätzlich auf einem Virtual Private Server (VPS) der HSR einen Jenkins Build-Server eingerichtet. Der Build-Server prüft mittels Polling ob neue Commits vorhanden sind und kompiliert gegebenenfalls jeweils eine neue Version der Applikation. So wird sichergestellt, dass eine Änderung nicht nur lokal kompiliert, sondern überall. Zudem bietet der Build-Server die Möglichkeit, die jeweils aktuellste Version der Applikation an Tester zu verteilen.

A.2. Projektmanagement

Zur Unterstützung des Projektmanagements wurde Redmine auf dem VPS der HSR eingerichtet. Vor allem die von Redmine gebotenen Funktionen Issue Verwaltung, Gantt Diagramm, Wiki und File-Repository wurden intensiv eingesetzt.

A.3. Testgeräte

Von der HSR wurden uns zwei Android Smartphones zur Verfügung gestellt. Ein „LG Nexus 5“ mit Android 6.0 sowie ein „Samsung Galaxy Nexus“ mit Android 4.3. Mit dem „Samsung Galaxy Nexus“ decken wir die Mindestanforderungen der Applikation (API Level 18 - Android 4.3) ab und mit dem „LG Nexus 5“ wird sichergestellt dass die Applikation auch auf Geräten mit der neusten Android Version (Stand Dezember 2015) läuft.

Zum simulieren der Küchengeräte erhielten wir von Fluxron einen Testaufbau. Dieser besteht aus drei Bluetooth Controllern wie sie bei Induktionsheizgeräten eingesetzt werden.

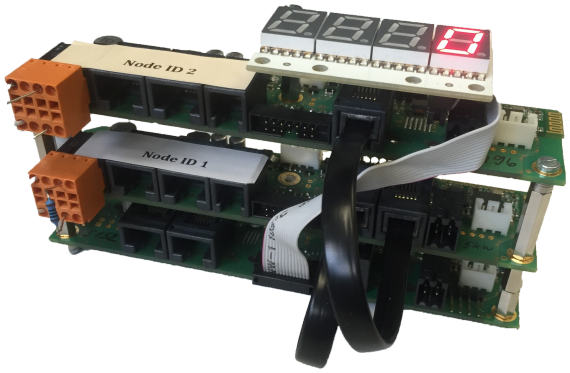


Abbildung A.1.: Fluxron Testaufbau

1. **Revision:** C-Class
   Typ: BAX
   Bluetooth: Classic

2. **Revision:** C-Class
   Typ: REX
   Bluetooth: Classic

3. **Revision:** C-Class
   Typ: BAX
   Bluetooth: Classic & 4.0

B. Lizenzvereinbarung

C. Detaillierte Systemtests

C.1. ST1: Küche initialisieren

Testziel Erfüllen von Use Case A. Der Benutzer erfasst die Lage der Geräte in der Küche.

Vorbedingungen

- Applikation gestartet
- Es sind mindestens 2 Geräte mit aktiver Bluetooth Verbindung vorhanden

Durchführung

1. Eine neue Küche erstellen und Namen vergeben
2. Das Layout der Küche nachstellen
3. Den Gerätesuchlauf starten
4. Die gefundenen Geräte im Küchenlayout platzieren
5. Die Küchenerstellung abschliessen

Erwartetes Ergebnis In der Applikation ist die neu erstellte Küche sichtbar und die platzierten Geräte zeigen ihren Status an.

C.2. ST2: Küche anpassen

Testziel Erfüllen von Use Case A. Der Benutzer kann die Initialisierung der Küche jederzeit wiederaufnehmen.

Vorbedingungen

- Applikation gestartet
- Eine initialisierte Küche ist bereits in der Applikation vorhanden
- Die Küche enthält mindestens 2 Geräte
- Die verwendeten Geräte haben eine aktive Bluetooth Verbindung

Durchführung

1. Die bestehende Küche öffnen
2. Ein Gerät auswählen und entfernen
3. Den Gerätesuchlauf starten
4. Das vorher entfernte Gerät soll gefunden werden und wieder zur Küche hinzugefügt werden
5. Die Küchenerstellung abschliessen

Erwartetes Ergebnis Die bestehende Küche hat nach der Durchführung dieses Tests die gleichen Geräte konfiguriert wie vorher.

C.3. ST3: Küche löschen

Testziel Erfüllen von Use Case A. Eine bereits erstelle Küche kann gelöscht werden.

Vorbedingungen

- Applikation gestartet
- Eine initialisierte Küche ist bereits in der Applikation vorhanden

Durchführung Eine bestehende Küche auswählen und löschen.

Erwartetes Ergebnis Die gelöschte Küche ist nicht mehr in der Applikation sichtbar.

C.4. ST4: Protokoll auslesen

Testziel Erfüllen von Use Case B. Der Benutzer kann das Protokoll eines Geräts auslesen.

Vorbedingungen

- Applikation gestartet
- Eine initialisierte Küche ist bereits in der Applikation vorhanden
- Die Küche enthält mindestens ein Gerät welches via Bluetooth erreichbar ist

Durchführung

1. Die bestehende Küche öffnen
2. Ein Gerät auswählen und dessen Protokollspeicher auslesen

Erwartetes Ergebnis Das Protokoll eines in der Küche vorhandenen Geräts wurde gelesen.

C.5. ST5: Geräteparameter ändern

Testziel Erfüllen von Use Case B2. Der Benutzer kann die Parameter eines Geräts verändern.

Vorbedingungen

- Applikation gestartet
- Eine initialisierte Küche ist bereits in der Applikation vorhanden
- Die Küche enthält mindestens ein Gerät welches via Bluetooth erreichbar ist

Durchführung

1. Die bestehende Küche öffnen
2. Ein Gerät auswählen und die Parameterliste dieses Geräts anzeigen
3. Einen Parameter wählen und einen neuen, gültigen Wert eingeben
4. Speichern

Erwartetes Ergebnis Die Parameter eines Geräts wurden verändert.

C.6. ST6: Geräteparameter ändern mit Validierung

Testziel Erfüllen von Use Case B2. Der Benutzer kann die Parameter eines Geräts verändern.

Vorbedingungen

- Applikation gestartet
- Eine initialisierte Küche ist bereits in der Applikation vorhanden
- Die Küche enthält mindestens ein Gerät welches via Bluetooth erreichbar ist

Durchführung

1. Die bestehende Küche öffnen
2. Ein Gerät auswählen und die Parameterliste dieses Geräts anzeigen
3. Einen Parameter wählen und einen neuen, ungültigen Wert eingeben
4. Speichern

Erwartetes Ergebnis Der Parameter akzeptiert den ungültigen Wert nicht. Es wird eine entsprechende Fehlermeldung dargestellt und die Änderung wird nicht gespeichert.

C.7. ST7: Kritische Geräteparameter ändern

Testziel Erfüllen von Use Case C. Mitarbeiter von Fluxron können sicherheitsrelevante Geräteparameter verändern.

Vorbedingungen

- Eine initialisierte Küche ist bereits in der Applikation vorhanden
- Die Küche enthält mindestens ein Gerät welches via Bluetooth erreichbar ist

Durchführung

1. Die Applikation starten
2. Sich als Mitarbeiter der Fluxron identifizieren
3. Die bestehende Küche öffnen
4. Ein Gerät auswählen und dessen sicherheitsrelevanten Parameter anpassen

Erwartetes Ergebnis Ein sicherheitsrelevanter Parameter eines Geräts wurde verändert.

C.8. ST8: Kennwortschutz

Testziel Erfüllen von FR1. Nur autorisierte Personen können die Applikation verwenden.

Vorbedingungen Die Applikation ist installiert.

Durchführung

1. Die Applikation starten
2. Passwort eingeben

Erwartetes Ergebnis Die Applikation lässt sich nach Eingabe des Passworts normal verwenden.

C.9. ST9: Kennwortschutz mit falschem Kennwort

Testziel	Erfüllen von FR1. Nur autorisierte Personen können die Applikation verwenden.
Vorbedingungen	Die Applikation ist installiert.
Durchführung	1. Die Applikation starten 2. Falsches Passwort eingeben
Erwartetes Ergebnis	Die Applikation zeigt eine Fehlermeldung und erlaubt eine erneute Eingabe des Passworts.

C.10. ST10: Suchlauf mit Gruppierung

Testziel	Erfüllen von FR2. Beim Suchlauf werden die gefundenen Geräte gruppiert dargestellt.
Vorbedingungen	• Applikation gestartet • Es sind mindestens 3 Geräte mit aktiver Bluetooth Verbindung vorhanden
Durchführung	Ein Gerätesuchlauf durchführen
Erwartetes Ergebnis	Die gefundenen Geräte werden übersichtlich gruppiert dargestellt.

C.11. ST11: Austausch von Küchenlayouts

Testziel Erfüllen von FR15. Bereits initialisierte Küchenkonfigurationen können mit anderen Benutzern ausgetauscht werden.

Vorbedingungen

- Ein Android Handy mit einer bereits initialisierten Küche
- Ein Android Handy ohne initialisierte Küchen

Durchführung

1. Auf einem Gerät die bestehende Küche öffnen
2. Die Küche mittels „Share-Button“ teilen
3. Auf dem zweiten Gerät kommt eine Benachrichtigung dass eine neue Küchenkonfiguration verfügbar ist
4. Diese Konfiguration akzeptieren und öffnen

Erwartetes Ergebnis Die Küchenkonfiguration vom einen Gerät ist nun auch auf dem zweiten Gerät verfügbar.

C.12. ST12: Küche mit grosser Anzahl Geräte (50)

Testziel Erfüllen von NFR1. Die Applikation soll zwischen 30-50 Fluxron Geräte problemlos verwalten können.

Vorbedingungen

- Applikation gestartet
- Es sind 50 Geräte (simuliert oder real) mit aktiver Bluetooth Verbindung vorhanden

Durchführung

1. Eine neue Küche initialisieren
2. 24 Geräte via Suchlauf finden und im Küchenlayout platzieren
3. Ein 25. Gerät hinzufügen und sich dessen Position/Name merke
4. Die restlichen 25 Geräte hinzufügen, so dass in der Küche jetzt total 50 Geräte vorhanden sind
5. die Küchen speichern
6. Gerät 25 in der Küche finden und dessen Status ansehen
7. Ein Parameter von Gerät 25 anpassen

Erwartetes Ergebnis Auch bei 50 Geräten sind auf einem spezifischen Gerät in der Küche noch alle Interaktionen möglich. Gerät 25 kann normal bedient werden.

C.13. ST13: Küche mit grosser Anzahl Geräte (150)

Testziel Erfüllen von NFR1. Die Applikation soll mit Einschränkungen bis zu 150 Fluxron Geräte verwalten können.

Vorbedingungen

- Applikation gestartet
- Es sind 50 Geräte (simuliert oder real) mit aktiver Bluetooth Verbindung vorhanden

Durchführung

1. Eine neue Küche initialisieren
2. 150 Geräte via Suchlauf finden und im Küchenlayout platzieren
3. Prüfen, dass der Status der Geräte sichtbar ist
4. Prüfen, dass einzelne Geräte noch auswählbar sind und deren Parameter verändert werden können

Erwartetes Ergebnis Bei 150 Geräten ist mit Einschränkungen zu rechnen. Die Applikation hat aber keine Performance-Probleme und es können alle kritischen Funktionen verwendet werden.

C.14. ST14: Bluetooth Unterstützung

Testziel Erfüllen von NFR5. Die Applikation kann sowohl mit Bluetooth Classic als auch Bluetooth 4.0 Geräten kommunizieren.

Vorbedingungen

- Aktives Fluxron Blutooth Classic Gerät
- Aktives Fluxron Blutooth 4.0 Gerät

Durchführung

1. Eine neue Küche initialisieren und Suchlauf starten
2. Die beiden Geräte hinzufügen

Erwartetes Ergebnis Der Status beider Geräte ist in der Küche sichtbar.

C.15. ST15: Keine Netzwerkkonnektivität

Testziel Erfüllen von NFR7. Die Applikation soll auch ohne aktive Internetverbindung voll funktionsfähig sein.

Vorbedingungen

- Applikation gestartet
- WLAN, GSM/UMTS/LTE deaktiviert
- Mindestens ein Gerät mit aktiver Bluetooth Verbindung vorhanden

Durchführung

1. Eine neue Küche initialisieren und Suchlauf starten
2. Das gefundene Gerät hinzufügen
3. Den Status des Geräts abrufen
4. Das Protokoll des Geräts abrufen
5. Die Küche löschen

Erwartetes Ergebnis Alle Aktionen sind möglich. Die Applikation verhält sich normal.

D. Arbeitspakete

Entsprechend zu den Meilensteinen wurden die folgenden Arbeitspakete geplant:

D.1. Analyse

Nr	Name	Inhalt
01	Ausgangslage	Kick-Off und Einarbeitung
02	Infrastruktur aufsetzen	VPS in Betrieb nehmen, Redmine installieren, Code und Dokumentation Repositories konfigurieren, Backup-Strategie erstellen.
03	Use Cases	Use Cases und User Stories anhand der Aufgabenstellung entwerfen und dokumentieren.
04	Anforderungsspezifikation	FR und NFR dokumentieren.
05	Testspezifikation	Aufgrund der FR und NFR eine Testspezifikation entwerfen.
06	Domainanalyse	Ein Domain-Modell der zu modellierenden Umgebung erstellen.
07	Meilensteine	Meilensteine erfassen und dokumentieren.

Tabelle D.1.: Arbeitspaket Analyse

D.2. Projektmanagement

Nr	Name	Inhalt
10	Arbeitspakete	Issues in Redmine erfassen und Arbeitspakete dokumentieren.
11	Projektplan	Projektplan erstellen, der die Arbeitspakete in einen zeitlichen Kontext bringt.
12	Risikomanagement	Risiken in Erfahrung bringen und dokumentieren.

Tabelle D.2.: Arbeitspaket Projektmanagement

D.3. Design

Nr	Name	Inhalt
20	Guidelines definieren	Coding und Style Guidelines definieren.
21	Architektur	Applikationsarchitektur definieren und begründen. Designentscheide und Patterns festhalten.
22	Evaluation Libraries	Libraries zur Unterstützung der Architektur evaluieren.
23	Nebenläufigkeitskonzept	Konzept zur Handhabung von Nebenläufigkeiten ausarbeiten.
24	UI Designstudie	Konzept für das Benutzerinterface entwerfen.
25	Architekturprototyp	Applikationsstruktur und Layers aufbauen. Eventbus einrichten. Via Bluetooth Werte auf dem Testgerät abfragen.

Tabelle D.3.: Arbeitspaket Design

D.4. Implementation

Beim implementieren der Applikation sind wir agil vorgegangen und haben mit Iterationen von jeweils einer Woche gearbeitet. Der Inhalt der einzelnen Iterationen wurde daher eine Woche vor dem Beginn der Phase konkretisiert und hier entsprechend nachgeführt.

Nr	Name	Inhalt
30	Implementation 0	Grundlayout für alle Activities. Layout Küchenliste. EDS Files einlesen und parsen. Fotofunktion für „Küche erstellen“. DeviceManager verwaltet Geräte.
31	Implementation 1	Grundlayout Küchenübersicht. Ankommende Nachrichten interpretieren. Checksummen-Prüfung von Nachrichten. Thumbnails von Bildern laden. Codegenerator für Parameter.
32	Implementation 2	Geräte platzieren und skalieren. Bluetooth Verbindungen cachen. Küchenbereich hinzufügen und speichern. Device Discovery UI. Geräte validieren.
33	Implementation 3	75% der Features sind implementiert. Device Activity mit Tabs. Konsistentes Navigationsdesign. Werte auf Gerät ändern können. Automatisiertes Pairing. Parameter zyklisch aktualisieren.
34	Implementation 4	Parameter in Device Ansicht darstellen. UI animieren. Layouts für Device Usage, Errors und Status. Custom Control zur Darstellung der Temperatur.
35	Implementation 5	Alle Features sind implementiert. Klassenerkennung bei Geräten. Schaltflächen stylen. Fixtexte konsequent im Ressourcen File hinterlegen. Error-Handling für Gerätefehler. Prozentzahlen für Nutzungsstatistik berechnen.

Tabelle D.4.: Arbeitspaket Implementation

D.5. Testing

Nr	Name	Inhalt
40	Bugfixes 0	Applikation mit lokalem Testaufbau testen. Kritische Bugs beheben. Sicherstellen, dass alle Muss-Features implementiert wurden.
41	Testtag bei Fluxron	Applikation mit echten Kochherden testen. Vorgehen gemäss der Systemtest Spezifikation.
42	Bugfixes 1	Restliche Bugs beheben. Design vereinheitlichen. Sicherstellen, dass der ganze Code mit JavaDocs versehen ist.

Tabelle D.5.: Arbeitspaket Testing

D.6. Dokumentation

Nr	Name	Inhalt
50	Zwischenpräsentation	Präsentation vorbereiten und halten.
51	Benutzerhandbuch	Benutzerhandbuch zur Applikation erstellt.
52	Abstract	Abstract geschrieben und bereit zur Abgabe.
53	Bericht	Dokumentation geschrieben und bereit zur Abgabe.
54	Schlusspräsentation	Präsentation vorbereiten und halten.

Tabelle D.6.: Arbeitspaket Dokumentation

E. Anleitung Weiterentwicklung

Die nachfolgenden Anleitungen dienen dazu, der Fluxron Solutions AG die Weiterentwicklung der Applikation zu erleichtern.

E.1. EDS Files aktualisieren

Wenn die Fluxron Geräte mit einer neuen Software Version neue Parameter erhalten, muss das „.eds File“ der jeweiligen Geräteklasse aktualisiert werden. Die „.eds Files“ befinden sich unter `/buildSrc/resources/ch.fluxron.generator/`.

Nachdem ein .eds File aktualisiert wurde, muss die Applikation neu gebuildet werden so dass die Klassen `DeviceParameter`, `ParamManager` sowie das Ressource File `parameters.xml` neu generiert werden. Nach dem Build stehen die neuen Parameter auch via Autocompletion zur Verfügung.

E.2. Neue Geräteklasse hinzufügen

Um eine neue Geräteklasse hinzuzufügen, muss zuerst das entsprechende „.eds File“ in den Ordner `/buildSrc/resources/ch.fluxron.generator/` kopiert werden. Danach kann in der Klasse `ParamGenerator` mit der Methode `loadParameter(..)` das neue .eds File angezogen werden. Zum Schluss sollte die Applikation neu gebuildet werden so dass die generierten Klassen und Ressource Files neu erstellt werden.

Für die neue Geräteklasse müssen nun die Layouts der 4 Fragments Status, Usage, Errors und Config erstellt werden. Dazu können die Layouts der bestehenden Geräteklassen kopiert und angepasst werden. Weitere Informationen zum Einbinden von Parametern bietet auch der Abschnitt E.3.

Zum Schluss muss noch in den Backing-Klassen der Layouts die neue Geräteklasse verwendet werden. Die Backing-Klassen befinden sich unter `app/java/ch.fluxron.fluxronapp/ui/fragments`. In der entsprechenden Klasse muss in der Methode `onCreateView` ein zusätzlicher Fall eingebaut werden welcher das neue Layout inflated.

E.3. Parameter in UI einbinden

Parameter können in den 4 Fragments (Status, Usage, Errors, Config) der Gerätedetailansicht eingebunden werden. Die dazugehörigen Layouts befinden sich unter `app/res/layout/`.

Als erstes muss also das entsprechende Fragment Layout geöffnet werden. Darin kann dann ein neues UI Control eingefügt werden. Für veränderbare Parameter sollte `ParameterEditable` verwendet werden, unveränderliche Parameter können als `ParameterView` oder `TemperatureBar` dargestellt werden. Die UI Controls verfügen über ein „ParamName“-Property, wo der gewünschte Parameter mittels Autocompletion gesetzt werden kann. `ParameterEditable` Controls verfügen zudem über ein „editableAccessLevel“-Property, womit die Sichtbarkeit

für die unterschiedlichen AccessLevels der User gesteuert werden kann.

Wenn das neue Control in einer bestehenden Liste von Parametern eingefügt worden ist, dann ist keine weitere Änderung mehr notwendig. Falls das Control ausserhalb eingebaut wurde, muss es in der entsprechenden Backing-Klasse noch angebunden werden. Die Backing-Klassen befinden sich unter `app/java/ch.fluxron.fluxronapp/ui/fragments`. In der entsprechenden Klasse muss beim Eintreten eines `DeviceChanged` Events die Methode `handleDeviceChanged(..)` auf dem Control des Parameters aufgerufen werden. Falls ein AccessLevel gesetzt wurde, muss ausserdem die Methode `handleAccessLevel(..)` beim Eintreten eines `AccessGranted` Events aufgerufen werden.

E.4. Neuen Meldungstyp einfügen

Um einen neuen Meldungstyp zu erstellen, muss eine neue Klasse im Package `ch.fluxron .fluxronapp.events` eingefügt werden. Für eine einfache Meldungsklasse ist dies bereits ausreichend. Für Meldungen, welche beantwortet werden sollen, empfiehlt es sich eine `RequestResponseConnection` zu verwenden. Diese gibt jeder Meldung eine Verbindungs-Id. Diese kann von anderen Meldungen übernommen werden. Damit können Antworten auf eine Meldung vom Absender der ursprünglichen Anfrage eindeutig identifiziert werden.

```
1  /**
2   * Requests the download of a news channel
3   */
4  public class DownloadNewsChannelCommand extends
       RequestResponseConnection {
5      private String newsChannelUrl;
6
7      /**
8       * New download
9       * @param newsChannelUrl Id of the kitchen
10      */
11     public DownloadNewsChannelCommand(String
       newsChannelUrl) {
12         this.newsChannelUrl = newsChannelUrl;
13     }
14
15     /**
16      * Returns the url of the Channel
17      * @return Url of the chennel
18      */
19     public String getNewsChannelUrl() {
20         return newsChannelUrl;
21     }
22 }
```

Abbildung E.2.: Neue Meldungsklasse mit einem übertragenen Wert

E.5. Empfangen und Senden von Meldungen

Die neue Meldungsklasse muss nun natürlich noch entgegengenommen werden. Dazu muss eine Klasse die Methode `onEventAsync(DownloadNewsChannelCommand)` besitzen und sich beim Event Bus registrieren.

```
1  /**
2   * Loads a news stream from the server
3   */
4  public class NewsDownloadManager {
5
6    private IEventBusProvider provider;
7
8    /**
9     * Creates a new news manager
10    * @param provider Provider
11    */
12   public NewsDownloadManager(IEventBusProvider provider) {
13       this.provider = provider;
14       provider.getUiEventBus().register(this);
15   }
16
17    /**
18    * Download requested
19    * @param msg Message with info
20    */
21   public void onEventAsync(DownloadNewsChannelCommand msg) {
22    NewsList latest = downloadNews(msg.getNewsChannelUrl());
23
24    NewsLoaded event = new NewsLoaded(latest);
25    event.setConnectionId(msg);
26    provider.getUiEventBus().post(event);
27   }
28 }
```

Abbildung E.3.: Abonnieren und Beantworten von Meldungen

Die Methode wird für jede empfangene Meldung aufgerufen und es kann darauf reagiert werden. Um das Ergebnis wieder zurückzuliefern, wird hier mit `setConnectionId(msg)` die Meldungs-Id des Commands wiederverwendet.

E.6. Synchrones warten auf Meldung

Achtung: Dies darf nur in `onEventAsync()` gemacht werden, ansonsten wird ein Deadlock erzeugt, da der aktuelle Thread blockiert wird.

In manchen Verarbeitungsprozessen ist es notwendig, z.B. zu warten bis eine Antwort von einem anderen Layer vorliegt. Dies ist meistens zwischen Businesslayer und Data Access Layer nötig. Nachfolgend ein Beispiel, in welchem auf das Speichern eines Objektes in der Datenbank gewartet wird. Somit bleiben die Managerklassen statuslos und können wie ein Webserver implementiert werden.

```
1 SaveObjectCommand saveCommand = new SaveObjectCommand();
2 saveCommand.setData(kitchenToSave);
3 saveCommand.setDocumentId("sampleId");
4
5 // Programm laeuft erst weiter, wenn eine Meldung mit
     derselben Verbindungs-Id empfangen wurde
6 WaitForResponse<RequestResponseConnection> waitForSave =
     new WaitForResponse<>();
7 RequestResponseConnection response =
     waitForSave.postAndWait(provider.getDalEventBus(),
     saveCommand, RequestResponseConnection.class);
```

Abbildung E.4.: Synchrones Warten auf Meldungen