HOCHSCHULE FÜR TECHNIK UND WIRTSCHAFT DRESDEN

Forschungsbericht im Themengebiet Sensornetze

Teilgebiet Web Frontend, Wintersemester 2013/14

Frank Effenberger (s69273), Rick Belitz (s69149)
Betreuer: Prof. Dr. Jörg Vogt
30.01.2014

Dieser Forschungsbericht ist eine überarbeitete Version des Berichtes des Sommersemesters 2013. Er beschäftigt sich mit der Konzeption eines prototypischen, anwenderfreundlichen Web Frontend für den Bereich der Hausautomatisierung. Als technologische Grundlage wird hierbei der FHEM-Server verwendet. Dabei finden neben einer Anforderungsanalyse technologische Betrachtungen und ein Entwurf, sowie eine prototypische Implementierung statt.

I. Inhaltsverzeichnis

II. Verzeichnis verwendeter Abkürzungen

6LowPan	IPv6 over Low power WPAN
A/E-Mode	Anfänger/Experten-Modus
Ajax	Asynchronous JavaScript and XML
Auth	Authentication
CAFM	Computer-Aided Facility Management
COAP	Constrained Application Protocol
CSS	Cascading Style Sheets
CUL	CC1101 USB Lite
DOM	Document Object Model
FHEM	Freundliche Hausautomatisierung und Energiemessung
FS	Funkschaltsystem
FTS4	Full Text Search Version 4
GUI	Graphical User Interface
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
HTW	Hochschule für Technik und Wirtschaft
HW	Hardware
IE	Internet Explorer
IPv6	Internet Protocol Version 6
JS	JavaScript
Lib	Library
Log	Logdatei
OSI (Modell)	Open Systems Interconnection Modell

PC	Personal Computer
PHP	Hypertext Preprocessor
RF	Radio Frequency
RFID	Radio-Frequency Identification
RWE	Rhein-Westfälisches Elektrizitätswerk AG
SQL	Structured Query Language
SS	Sommersemester
SVG	Scalable Vector Graphics
WPAN	Wireless Personal Area Network
WS	Wintersemester
WWW	World Wide Web
YAF	Yet Another Floorplan
XML	Extensible Markup Language

III. Glossar

Aktor	In diesem Kontext ist damit der technische Begriff eines Aktors zu verstehen. Ein solcher Aktor wandelt elektrische Signale (z.B. Computerbefehle) in physikalische oder chemische Größen um (z.B. mechanische Bewegung, Erhöhung der Luftfeuchte)[1].
Array	Ein Array ist ein Datentyp, der mehrere gleichartige Daten beinhalten kann.
Bibliothek	In diesem Kontext wird unter einer Bibliothek ein Framework und Sammlung von Programmen verstanden, das zum Programmieren genutzt werden kann.
Bluff	Dies ist eine Open Source JS-Charting Bibliothek.
Caching	Der Vorgang des Zwischenspeicherns von websiterelevanten Daten, um beim wiederholten Aufruf der Seite kürzere Antwortzeiten zu haben.
Canvas 3D Graph	Dies ist eine Open Source JS-Charting Bibliothek (nur 3D).
CanvasXpress	Dies ist eine Open Source JS-Charting Bibliothek.
Ccchart	Dies ist eine Open Source JS-Charting Bibliothek (via Websockets).
Charting	Damit ist die Erstellung von Charts (Diagrammen) gemeint.
Client-/Serverlast	Diese Begrifflichkeit wird in diesem Kontext als Abwägung zwischen der Verarbeitung und Bereitstellen von Daten und Grafik auf Client- oder auf Serverseite verwendet.

[1] In Anlehnung an ([WIKI13] Aktor, 2013), http://de.wikipedia.org/wiki/Aktor.

CUL	Dies ist ein USB-Funksender.
D3	Dies ist eine Open Source JS-Charting Bibliothek.
dhtmlxChart	Dies ist eine Open Source JS-Charting Bibliothek.
Document Object Model	Eine Spezifikation des World Wide Web Consortiums[2], die unter anderem in Webseiten ihre Anwendung findet.
dygraphs	Dies ist eine Open Source JS-Charting Bibliothek.
Elycharts (jQuery)	Dies ist eine Open Source JS-Charting Bibliothek.
EZcontrol	Damit ist Hausautomatisierungslösung der EZcontrol GmbH gemeint.
FHEM-Server	In Perl geschriebener Server als Open Source Lösung für die Hausautomatisierung.
Floorplan	Ein Grundriss, der im Rahmen der Hausautomatisierung um Funktionalitäten erweitert wird.
Flot (jQuery)	Dies ist eine Open Source JS-Charting Bibliothek.
Flotr2	Dies ist eine Open Source JS-Charting Bibliothek.
Frontend (Web)	Dies ist die Nutzersicht auf eine Anwendung (z.B. eine Internetseite als Webfrontend).
FS20	Dies das Funkschaltsystem FS20, zu dem es mehrere Produktarten gibt.
FusionCharts	Auf Dashboards spezialisierte, proprietäre JS-Charting Bibliothek.
Gnuplot	Programm zur grafischen Darstellung von diversen Daten.

[2] Vgl. hierzu ([LHÈG05] Philippe Le Hégaret, 2005), http://www.w3.org/DOM/.

Google Charts	Dies ist eine Open Source JS-Charting Bibliothek.
gRaphaël	Dies ist eine Open Source JS-Charting Bibliothek.
Hash	Dies ist eine Streuwertfunktion.
Hausautomatisierung	Ein Teilgebiet der Gebäudeautomation, mit Fokus darauf, Vorgänge im alltäglichen Leben mithilfe von Sensoren, Aktoren und einer Steuersoftware zu automatisieren.
Highcharts	Dies ist eine proprietäre JS-Charting Bibliothek.
Homematic	Damit sind die Hardwarelösungen zur Hausautomatisierung von eQ-3 AG gemeint.
Homeputer	Damit ist die Softwarelösung zur Hausautomatisierung von eQ-3 AG gemeint.
Ico	Dies ist eine Open Source JS-Charting Bibliothek.
Interface	Dies ist eine Schnittstelle einer Anwendung.
Internet der Dinge	Zusammenfassender Begriff des Themas der wachsenden Anzahl der Geräte mit Befähigung, über das Internet zu kommunizieren.
IP-Symcon	Damit ist die Softwarelösung zur Hausautomatisierung der Symcon GmbH gemeint.
jqPlot (jQuery)	Dies ist eine Open Source JS-Charting Bibliothek.
jQuery	Eine JavaScript-Bibliothek zur Manipulation des DOM.
JS Charts	Dies ist eine proprietäre JS-Charting Bibliothek.

JSXGraph	Dies ist eine Open Source JS-Charting Bibliothek.
Liste	Ein Datentyp, bei dem gleichartige Daten in einer Folge verkettet gespeichert werden können.
Look and Feel	Eine Umschreibung für die nichtfunktionalen Anforderungen ‚Selbstbeschreibungsfähigkeit' und ‚Benutzbarkeit'.
Marmitek	Damit ist die Softwarelösung zur Hausautomatisierung von Marmitek BV gemeint.
MilkChart (MooTools)	Dies ist eine Open Source JS-Charting Bibliothek.
MouseOver	Bezeichnung des Ereignisses, wenn der Mauszeiger über ein Objekt fährt.
Namespace	Ein Namensraum zur Qualifizierung und Identifikation von Elementen.
Neues Charting Frontend	Die aktuelle Charting-Tool Lösung der FHEM-Community.
Open Hardware	Damit ist das Feld der freie Hardware gemeint.
Open Source	Bestimmung, der der Quellcode einer Anwendung für jeden sicht- und nutzbar ist.
Pairing	Ein Vorgang, bei dem ein Aktor oder Sensor von einem anderem System erkannt wird.
Perl	Perl ist eine plattformunabhängige Programmiersprache.
PGM 2	Standard Webfrontend des FHEM-Servers
PGM 3	Ein auf PHP-basierendes Webfrontend für den FHEM-Server.

PGM 5	Ein auf Perl basierendes Webfrontend des FHEM-Servers, ausgelegt auf niedrige Antwortzeiten und hohe Effizienz bei der Bildgenerierung für leistungsschwache Systeme.
Plotkit (Mochikit)	Dies ist eine Open Source JS-Charting Bibliothek.
Plotting	Der Begriff wird in diesem Kontext als Synonym für Charting verwendet.
Prototyp	In diesem Kontext ist damit eine Vorab-Version der Softwarelösung gemeint, welche nur begrenzte oder keine Funktionalität (nur Visualisierung) bietet.
Protovis	Dies ist eine Open Source JS-Charting Bibliothek.
Regelverarbeitung	In diesem Kontext ist damit die Erstellung von Regeln zur Steuerung von Aktoren im FHEM-Server gemeint.
RWE Smarthome	Dies ist die Hausautomatisierungsproduktreihe von RWE.
Sensor	Ein Sensor ist ein technisches Gerät, welches Größen (z.B. Luftfeuchte) messen kann.
Smartphone	Ein Mobiltelefon, welches zusätzliche Funktionalitäten als einfache Sprach- oder Nachrichtenübermittlung bietet, z.B. Touchscreen oder umfangreiche Applikationen.
SQLite-Datenbank	Eingebettetes Datenbanksystem (Programm-Bibliothek) mit grundlegenden SQL-Funktionen.
Tablet	Ein Tablet ist ein Computer in flacher, handlicher und tragbarer Form, in der Regel kombiniert mit einem Touchscreen und eingeschränkter Funktionalität.

Tag	Dies ist ein Auszeichnungselement.
Tooltip	Dies sind Popups, welche Hilfetexte oder Beschreibungen bereitstellen.[3]
Use Case	Ein Anwendungsfall, in der ein Akteur (ggf.) auf verschiedene Art und Weisen versucht, ein Ziel zu erreichen.

[3] Vgl. hierzu ([HAPK05 S.16] Hapke, 2005).

IV. Abbildungsverzeichnis

1. Einleitung

Dieser Forschungsbericht ist eine überarbeitete Version des Berichtes des Sommersemesters 2013. Er beschäftigt sich mit der Konzeption eines prototypischen, anwenderfreundlichen Web Frontend für den Bereich der Hausautomatisierung. Als technologische Grundlage wird hierbei der FHEM-Server verwendet. Es finden neben einer Anforderungsanalyse technologische Betrachtungen und ein Entwurf, sowie eine prototypische Implementierung statt.

1.1 Motivation

Das heutige Zeitalter wird von wachsender Mobilität, Erreichbarkeit und höheren Datenraten im Bereich des mobilen Internets geprägt[4]. So ist auch in Deutschland eine steigende Nachfrage an Netzabdeckung gegeben[5]. Dies ist zum einen dadurch bedingt, dass mehr Nutzer neben ihrem PC mittlerweile einen Laptop und Smartphone benutzen, zum anderen können mehr Geräte über das Internet kommunizieren als zuvor. Dies beinhaltet auch Sensoren oder Pakete, welche beispielsweise über RFID-Chips geortet werden und deren Ort und Zustände über das Internet abgefragt werden können. Diese Häufung der Kommunikation von Geräten und Sensoren im Internet hat zur Prägung eines neuen Themas geführt: Das Internet der Dinge[6].

Dieses Thema spielt neben der Paketverfolgung auch eine Rolle im Gesundheitswesen[7], der Werkstoffprüfung[8], der Industrieautomation[9] und in der Hausautomatisierung[10]. Letztere ist vor allem dadurch geprägt, dass Sensoren und Aktoren einen starken Einfluss auf das private Umfeld der Personen haben können und sollen.

Das Forschungsseminar ‚Sensornetze' greift diese aktuelle Thematik des Internets der Dinge auf und beschäftigt sich mit der Kommunikation von Sensoren und Aktoren über das Web. Dabei fand in den letzten Jahren eine Fokussierung auf den Bereich der Hausautomatisierung im Open Source-Bereich statt. Da es in diesem Teilgebiet jedoch noch keine umfassende Anforderungsanalyse und eine für programmierunerfahrene Nutzer nutzbare Weboberfläche gibt, beschäftigt sich diese Teilgruppe mit der Konzipierung und prototypischen Implementierung eines Web Frontend.

[4] Vgl. hierzu ([HUUV13] Meldung der Bundesnetzagentur über 325 Millionen umts Nutzer im Jahr, 2013).
[5] Vgl. hierzu ([HUUV10] Halbjahresbericht 2010 zur UMTS-Abdeckung, 2010).
[6] Vgl. hierzu ([FIML13a] Frauenhofer Institut für Materialfluss und Logistik zum Internet der Dinge, 2013).
[7] Vgl. hierzu ([FIFF13a] Frauenhofer Institut für Fabrikbetrieb und ~automatisierung, 2013).
[8] Vgl. hierzu ([FIFF13b] Frauenhofer Institut für Fabrikbetrieb und ~automatisierung, 2013).
[9] Vgl. hierzu ([FIML13b] Frauenhofer Institut für Materialfluss und Logistik, 2013).
[10] Vgl. hierzu ([FHEM13] Homepage für freundliche Hausautomatisierung und Energiemessung, 2013).

1.2 Zielstellung

Zielstellung des Teilgebietes ‚Web Frontend' ist es, ein nutzerfreundliches Web Frontend für die Hausautomatisierung zu konzipieren und prototypisch zu implementieren. Als Grundlage wurde dazu von Prof. Dr. Vogt der FHEM-Server vorgegeben.

Dabei gilt es zeitgleich, Antworten auf folgende Fragen zu finden:

- *Welche besonderen funktionalen sowie nichtfunktionalen Anforderungen gelten im Bereich der Hausautomatisierung?*

- *Welche dieser Anforderungen lassen sich mit einem Web Frontend realisieren?*

- *Welche Eigenschaften hat eine nutzerfreundliches Web Frontend unter Berücksichtigung verschiedener Nutzertypen im Bereich der Hausautomatisierung?*

Ziel des Forschungsberichtes im Wintersemester 2013/14 ist es, Antworten auf diese Fragen zu finden, sowie eine Analyse durchzuführen und einen Entwurf für ein Web Frontend im Bereich der Hausautomatisierung zu konzipieren. Die prototypische Implementierung ist ebenso Ziel dieses Semesters.

1.3 Lösungsweg

Im Rahmen des Forschungsseminares gab es wöchentliche Treffen, in denen jede der insgesamt vier Gruppen ihre Ergebnisse präsentierten und eine anschließende Diskussion stattfand. Auf Grundlage dieser Diskussion und dem bisherigen, iterativem Vorgehen wurde ein Lösungsweg zur Beantwortung der Fragen aus der Zielstellung (siehe *1.2 Zielstellung)* erarbeitet.

Zu Beginn fand eine intensive Betrachtung des gegebenen FHEM-Servers statt. Dabei wurden Technologien und sein Aufbau ebenso wie Anbindungen von Web Frontends untersucht, sowie Testdaten angelegt und ein echter Sensor zur Outdoor-Messung von Temperatur und Luftfeuchte in der HTW Dresden installiert.

Im Anschluss wurde eine Anforderungsanalyse durchgeführt, welche zuerst nichtfunktionale Anforderungen sammelte und zusammen mit einer Risikoanalyse allgemeine, funktionale Anforderungen herzuleiten. Zusätzlich sind dabei Anforderungen von der FHEM-Community aufgenommen worden. Zeitungsberichte, Fernsehsendungen, Befragungen und Betrachtung bisheriger System waren ebenso eine Quelle für die Generierung der funktionalen Anforderungen. Dieser Anforderungskatalog befindet sich nicht in einem vollständigen Stadium, er soll und kann in Zukunft erweitert werden. Anhand dieser Anforderungen wurden beispielhaft Anwendungsfälle unterschiedlicher Komplexität textuell beschrieben, um daraus aus einer zukünftigen Bedienung des Web Frontend schließen zu können.

Anhand des allgemeinen, funktionalen Anforderungskataloges und der exemplarischen Anwendungsfälle wurden nun allgemeine Anforderungen für ein Web Frontend abstrahiert und modelliert. Zusätzlich zur Betrachtung bisheriger, meist proprietärer Lösungen im Bereich der Hausautomatisierung fanden auch technologische Betrachtungen für eine mögliche, prototypische Implementierung statt. All dies bildet die Grundlage für den danach erstellten Entwurf.

Auf Basis des Entwurfes fand eine prototypische Implementierung der Anforderungen statt, welche mit der höchsten Priorität durch die Teilgruppe Web Frontend versehen wurde.

2. Kurzbetrachtung des FHEM-Server

Im vergangenen Forschungsseminar wurde der FHEM-Server als Serverzentrale zum Ansprechen der Aktoren und Sensoren ausgewählt, installiert und ist momentan in Verwendung. Da das Konzept des eigenen Web Frontend Funktionalitäten des FHEM-Servers nutzen muss, findet eine Kurzbetrachtung des FHEM-Servers und seiner für diesen Teilbereich wichtigen Eigenschaften statt.

2.1 Der FHEM-Server

„FHEM ist ein in perl geschriebener, GPL lizensierter Server für die Heimautomatisierung. Man kann mit FHEM häufig auftretende Aufgaben automatisieren, wie z.Bsp. Lampen / Rollladen / Heizung / usw. schalten, oder Ereignisse wie Temperatur / Feuchtigkeit / Stromverbrauch protokollieren und visualisieren (...)“

„(...) Das Programm läuft als Server, man kann es über WEB, dedizierte Smartphone Apps oder telnet bedienen, TCP Schnittstellen für JSON und XML existieren ebenfalls.

Um es zu verwenden benötigt man einen 24/7 Rechner (Fritz!Box, NAS, RPi, PC, MacMini, etc) mit einem perl Interpreter und angeschlossene Hardware-Komponenten wie CUL, FHZ1300PC, etc. für einen Zugang zu den Aktoren und Sensoren.(...)“[11]

2.1 Kommunikationsschnittstelle via Telnet

Für die spätere Konzeption ist eine Kommunikationsschnittstelle zum Absetzen von FHEM-spezifischen Befehlen von elementarer Bedeutung. Dabei ist es möglich, über Telnet (Port 7072) eine Verbindung zu einem installierten FHEM-Server aufzusetzen und Befehle abzuschicken und eine Antwort zu erhalten. Dies bildet ebenso die Grundlage für eines der bestehenden Web Frontends.

[11] Vgl. hierzu ([KÖNI13] König - Website von FHEM, 2013).

2.2 Für FHEM vorhandene Web Frontends

Für den FHEM-Server gibt es bereits mehrere, bereits vorhandene Web Frontends, welche die Bezeichnungen PGM 2, PGM 3 und PGM 5 tragen. Es gibt auch Ordner für PGM und PGM 4, die allerdings entweder leer oder mit nicht verwendbaren Dateien gefüllt sind, so dass diese nicht in der folgenden Kurzvorstellung betrachtet werden.

PGM 2:

Dies ist das standardmäßige, von FHEM mitgelieferte Web Frontend. Es ist in Perl geschrieben und nutzt den kleinen http-Server, der bei der FHEM-Installation mit konfiguriert wird. Dieses Web Frontend bietet umfassende Funktionalitäten, ist jedoch von der Navigation, Gliederung und dem Einsatz der Technologien her verbesserungswürdig.

PGM 3:

Dieses Web Frontend basiert auf PHP und benötigt einen eigenen Server, um auszuführen. Befehle werden über Telnet auf Port 7072 dann an den FHEM-Server geschickt, die Antworten entgegengenommen und verarbeitet. Die hier verwendeten Technologien, Optik und Navigation sind dabei nicht an momentane Standards oder neueste Technologien angelehnt.

PGM 5

Dieses Web Frontend ist ebenso in Perl geschrieben und nutzt dafür das CGI-Modul und CSS zur Gestaltung der Website. Dieses Web Frontend ist auf eine Minimierung der CPU-Last auf Serverseite ausgelegt und bietet daher auch optisch keine ansprechenden Lösungen und ist für leistungsschwache Server ausgelegt.

3. Anforderungsanalyse

Zur Konzipierung eines Web Frontend und zur Beantwortung der Fragen der Zielstellung (siehe *1.2 Zielstellung*) ist ein Anforderungskatalog nötig. Daher wurden zuerst nichtfunktionale Anforderungen gesammelt und eine Risikoanalyse zur Vermeidung von Fehlern durchgeführt. Danach wurden allgemeine, funktionale Anforderungen ermittelt und in den Anforderungskatalog aufgenommen. Exemplarisch wurden dazu textuell Anwendungsfälle unterschiedlicher Komplexität formuliert, anhand derer zusammen mit den allgemeinen, funktionalen Anforderungen nun abstrahierte Anforderungen speziell für ein Web Frontend extrahiert werden konnten.

3.1 Nichtfunktionale Anforderungen

Bevor mit der Sammlung von funktionalen Anforderungen begonnen wurde, mussten nichtfunktionale Anforderungen gesammelt werden. Dazu wurden neben der reinen Nennung der Anforderung auch eine nähergelegene Beschreibung und eine Lösung zur Einhaltung der Anforderung notiert. Im Folgenden nun ein Auszug der Tabelle (vgl. **Tabelle 1: Nichtfunktionale Anforderungen (Auszug)**) mit nichtfunktionalen Anforderungen, welche im Anhang (siehe *Anlage 1: Nichtfunktionale Anforderungen*) zu finden ist.

Tabelle 1: Nichtfunktionale Anforderungen (Auszug)

Anforderung	Kurzbeschreibung	Lösung
Sicherheit	Maßnahmen zur Vermeidung von gefährlichen Zuständen.	Authentifizierung, Verschlüsselung.
Funktionserfüllung	Übereinstimmung funktionale Anforderung und realisierte Funktionalität.	Testen der Anwendung.
Benutzbarkeit(Usability)	Alle Eigenschaften, die dem Benutzer ein angenehmes, einfaches und effizientes Arbeiten ermöglichen.	Klare Strukturierung und Komposition der Website.
Effizienz	Maß für die Inanspruchnahme von Betriebsmitteln bei gegebenem Funktionsumfang.	Kombination von PHP und JavaScript.
Effektivität	Maß für die Wirksamkeit, Verhältnis von erreichtem zu definiertem Ziel.	Regelmäßiges Überprüfen der realisierten Funktionalität in Hinblick auf Anforderungsanalyse.

3.2 Risikoanalyse

Da während der Anforderungsanalyse Fehler unterlaufen können, macht es Sinn, eine Risikoanalyse durchzuführen und Präventivmaßnahmen anzuwenden, um die Qualität des Anforderungskataloges zu gewährleisten. Im Folgenden nun die Tabelle (vgl. **Tabelle 2: Risikoanalyse**), in der die Risiken, ihre Gegenmaßnahmen und Qualitätsmerkmale zu finden sind.

Tabelle 2: Risikoanalyse

Risiko	Risikovermeidung	Qualitätsmerkmal
Es wurden falsche Anforderungen ermittelt.	Diskussion mit Nutzern, Oberflächenprototyp.	Korrektheit
Die Anforderungen wurden unvollständig ermittelt.	Diskussion mit Nutzern. Erweiterung des Kataloges bei Bedarf.	Vollständigkeit
Die Anforderungen sind missverständlich beschrieben.	Wahl von Darstellung, die Eindeutigkeit unterstützt, Diskussion mit Nutzern.	Eindeutigkeit
Die ermittelten Anforderungen sind widersprüchlich beschrieben.	Wahl von Darstellungstechnik und Werkzeugen, die die Konsistenz unterstützen.	Widerspruchsfreiheit (Konsistenz)
Die Prioritäten (Wichtigkeit für den Nutzer) werden falsch gesetzt.	Diskussion mit Nutzer.	Angemessenheit
Die Anforderungen sind umständlich und zu umfangreich.	Wahl von Darstellungstechnik, die Kürze und Prägnanz unterstützen.	Minimalität
Die Anforderungen sind nicht realisierbar.	Experimenteller Prototyp (testen, ob man es realisieren kann).	Realisierbarkeit

3.3 Allgemeine, funktionale Anforderungen

Anhand der nichtfunktionalen Anforderungen konnte unter zusätzlicher Berücksichtigung der Risikovermeidungsstrategien nun begonnen werden, allgemeine Anforderungen zu sammeln. Dabei wurden Anforderungen anhand eigener Erfahrungen, Diskussionen mit Mitmenschen (in und außerhalb des Forschungsseminares), der FHEM-Community[12], Zeitungsartikeln[13], Fernsehsendungen und Internetseiten[14,15] gesammelt. Im Folgenden nun ein Auszug der Tabelle (vgl. **Tabelle 3: Allgemeine, funktionale Anforderungen (Auszug)**) mit den allgemeinen, funktionalen Anforderungen. Die gesamten Anforderungen sind im Anhang (siehe *Anlage 2: Allgemeine, funktionale Anforderungen*) zu finden.

Tabelle 3: Allgemeine, funktionale Anforderungen (Auszug)

Nr	Name der Anforderung	Beschreibung	Merkmale	Sonstiges (Einordnung)
1	Steuerung der Rolläden/ Jalousien via Web Frontend.	Nutzer kann via Endgerät die Rollläden/Jalousien hoch- und runterfahren lassen.	Stufenweises herunter/hochfahren sollte möglich sein.	Steuerung
2	Steuerung der Rolläden/ Jalousien über Aktoren/ Sensoren.	Rollläden/Jalousien reagieren bei bestimmten Sensorwerten.	Gruppensteuerung mehrerer Rollläden/Jalousien, Zeitsteuerung, Sonneneinstrahlung messen.	Steuerung / Regelverarbeitung
3	Lichtsteuerung via Web Frontend.	Nutzer kann via Endgeräte die Lichter steuern.	Schalten, dimmen	Steuerung
4	Lichtsteuerung via Aktoren/ Sensoren.	Licht kann bei bestimmten Sensorwerten gesteuert werden.	Zeitsteuerung, Bewegungsmelder, dimmen, schalten	Steuerung / Regelverarbeitung
5	Heizungs-steuerung via Web Frontend.	Nutzer kann via Endgeräte die Heizung steuern.	Schalten, Temperieren	Steuerung (auch Fußbodenheizung)
6	Heizungs-steuerung via Aktoren/ Sensoren.	Heizung kann bei bestimmten Sensorwerten gesteuert werden.	Schalten, temperatur- und zeitabhängiges Temperieren	Steuerung / Regelverarbeitung (auch Fußbodenheizung), wenn gelüftet wird Heizung ausschalten.

[12] Vgl. hierzu ([EFFE13] Frank Effenberger - Diskussion im FHEM-Forum, 2013).
[13] Vgl. hierzu ([BERN13] Henry Berndt - Guck mal wer da spricht - Sächsische Zeitung, 2013).
[14] Vgl. hierzu ([RWE13], RWE Smarthome Website, 2013)
[15] Vgl. hierzu ([KALIE13] Andy Kalies Beispiele für Hausautomatisierung)

3.3.1 Anwendungsfälle (Beispiele)

Anhand der allgemeinen, funktionalen Anforderungen wurden exemplarisch einige Anwendungsfälle unterschiedlicher Komplexität beschrieben, um sich so bereits vorab über eventuelle Abläufe bei der Nutzung des Web Frontend Gedanken machen zu können.

Beispiel eines **einfachen Anwendungsfalles**:

Der Nutzer möchte seine Dimmlampe (mit Aktor) mithilfe des Web Frontend registrieren. Danach möchte er die Dimmlampe regulieren.

1. Anmelden als Administrator
2. Anlegen des Gerätes im administrativen Bereich (Steuerung)
3. Pairingverfahren durchführen (Ergebnis: Erkennung des Gerätes oder Fehler)
4. Zuordnen des Gerätes zu einem Raum
5. Wechseln zum Bereich ‚Home'
6. Manuelle Regulierung der Lampe (z.B. Klick, Dropdown-Menü, Slider oder Zahlenwert).

Beispiel eines **mehrschichtigen Anwendungsfalles**:

Der Nutzer möchte einen Sensor und Aktor seiner Heizung mithilfe des Web Frontend registrieren. Danach möchte er die Heizung über eine Regelverarbeitung uhrzeitmäßig steuern. Anschließend möchte er eine Analyse, sowie Auswertung der Sensordaten durchführen.

1. Anmelden als Administrator
2. Anlegen des Gerätes im administrativen Bereich (Steuerung)
3. Pairingverfahren durchführen (Ergebnis: Erkennung des Gerätes oder Fehler)
4. Zuordnen des Gerätes in einen Raum
5. Erstellung einer uhrzeitabhängigen Regel für die Heizung
6. Wechseln zum Bereich ‚Home'
7. Ansehen der gespeicherten Sensordaten für die Heizung durch Klick auf Sensor im Raum
8. Auswertung anhand der Statistik (generiertes Chart)

Beispiel eines **komplexen Anwendungsfalles**:

Der Nutzer möchte seine Dimmplampe (Aktor) sowie einen Bewegungsmelder (Sensor) mithilfe des Web Frontend registrieren. Anschließend möchte er über eine Regelverarbeitung bei Meldung einer Bewegung die Lampe aktivieren. Die Leuchtkraft der Lampe soll dabei uhrzeitabhängig sein. Im Anschluss möchte der Nutzer eine Statistik einsehen, wie oft die Lampe im Mittel am Tag eingeschaltet war und wie hoch der Stromverbrauch ist.

1. Anmelden als Administrator in der GUI
2. Anlegen der Geräte im administrativen Bereich
3. Pairingverfahren durchführen (Ergebnis: Erkennung des Gerätes oder Fehler)
4. Zuordnen der Geräte zu einem Raum (mehreren Räumen)
5. Erstellung einer sensorabhängigen Aktivierung der Dimmlampe, sowie einer uhrzeitabhängigen Regulierung der Leuchtkraft
6. Wechseln zum Bereich ‚Home‘
7. Ansehen der gespeicherten Sensordaten für die Dimmplampe und Stromverbrauch durch Klick auf den Sensor
8. Auswertung anhand der Statistiken (generiertes Chart)

3.4 Abstrahierte, funktionale Anforderungen

Anhand der Anwendungsfälle und der allgemeinen, funktionalen Anforderungen konnten nun abstrahierte, funktionale Anforderungen für das Web Frontend generiert werden. Dabei erfolgte eine Priorisierung (vgl. **Abbildung 1: Interaktion des Endnutzers mit dem System**), wobei eine niedrige Zahl eine höhere Priorität (im Bild rot markiert) hat als eine hohe Zahl (im Bild grün markiert). Da das gesamte Forschungsseminar auf ein Jahr beschränkt ist, mussten anhand dieser Priorisierung die Funktionen herausgegriffen werden, welche als erstes prototypisch implementiert werden. Die komplette Auflistung der Anforderungen und ein weiteres Diagramm aus Administrationssicht ist im Anhang (siehe *Anlage 3: Abstrahierte, funktionale Anforderungen*) zu finden.

Abbildung 1: Interaktion des Endnutzers mit dem System

4. Technologische Betrachtungen

Nachdem die abstrahierten, funktionalen Anforderungen ermittelt wurden, muss nun noch abgewogen werden, welche Programmiersprache gewählt wird, ob die Lastverteilung eher auf Client- oder auf Serverseite liegt und welche Bibliotheken für das Charting verwendet werden. Die Betrachtung bereits vorhandener (und zumeist) proprietärer Systeme ist dabei ebenso eine Hilfestellung.

4.1 Abwägungen zwischen Perl und PHP

Im Rahmen dieses Abschnittes wird eine Abwägung der Programmierung in Perl und PHP vollzogen. Dabei gilt als Vorbetrachtung, dass lediglich ein Web Frontend erstellt werden soll und alle weiteren Funktionalitäten durch den angebundenen FHEM-Server übernommen werden.

Da der FHEM-Server bereits in Perl geschrieben wurde, bietet es sich an, das Web Frontend selbst direkt in Perl zu schreiben und anzubinden. Zeitgleich ist das Erlernen dieser Sprache recht schwer und moderne Technologien wie Ajax, CSS3, HTML5 lassen sich im Vergleich zu PHP eher umständlich integrieren und für künftige Generationen warten. Zur Abwägung wurde folgende Tabelle (vgl. **Tabelle 4: Vor- und Nachteile von Perl**) erstellt (grün ist positiv, rot ist negativ):

Tabelle 4: Vor- und Nachteile von Perl

Vorteile PERL: (PGM2 oder PGM5):	Nachteile PERL (PGM2 oder PGM5):
Es gibt sehr weitreichende Möglichkeiten durch Module (Sensor ansteuern, Pairing, etc.).	Es ist eine lange Einarbeitungszeit nötig.
Es ist eine gute Arbeit mit Arrays, Listen, Hashes möglich.	Zu Teilen sehr schwer verständlicher Code im FHEM-Server und in den Modulen.
Perl ist schneller als PHP bei übergreifender Nutzung verschiedener Technologien und Sprachen in einem Programm.	Wer wartet das Web Frontend und arbeitet damit nach dem Seminar? Diese Frage war im Rahmen der Wartbarkeit nicht klärbar.
Es ist Arbeit mit Namespaces möglich, Perl ist zudem eine kontextsensitive Sprache.	Webseitengenerierung deutlich kryptisch und schwierig (Modul: CGI als Grundlage).
	Sehr mächtige Sprache, aber es wird nur ein Webfrontend benötigt.
	Im Gegensatz zu PHP deutlich seltener verwendet als Webauftritt.
	Viele Webhoster haben kein Perl mehr.
	Die Sprache Perl ist nicht zum Erstellen von Webseiten entwickelt wurden.
	Zukunftsfähigkeit für Websiten ungeklärt.

Während Perl durch eher negative Merkmale zur Erfüllung dieser Aufgabe aufgefallen ist, folgt nun (vgl. **Tabelle 5: Vor- und Nachteile von PHP**) eine Abwägung gegenüber PHP (grün ist positiv, gelb ist zu bedenken):

Tabelle 5: Vor- und Nachteile von PHP

Vorteile PHP:	Nachteile PHP:
PHP-Dateien enthalten verständlicheren Quellcode als Perlskripte, zudem ist eine gute Einarbeitung möglich.	Es ist ‚nur' für den Webteil nutzbar. Das Ansteuern von Sensoren ist mit PHP unmöglich (Aber: Man kann auf die Perl-Methoden vom FHEM-Server zugreifen und damit alles auch nutzen).
Nachfolger können mit kurzer Einarbeitungszeit am Thema weiterarbeiten.	
PHP bietet moderne Technologien zur Webseitenerstellung und Bildgenerierung (JS, HTML5, AJAX). Diese sind sehr gut und einfach nutzbar.	
Mithilfe eines PHP-Sockets auf Port 7072 lassen sich die FHEM-Befehle und Funktionalitäten nutzen.	
Bietet viele komfortable Grundfunktionen und wird ständig erweitert.	
PHP deckt benötigte Funktionalität vollständig ab.	
Die meisten Internetauftritte sind in PHP, es gibt eine große, hilfsbereite Community.	
PHP wurde von Beginn an für die serverseitige Webseitenerstellung entwickelt.	
Es ist eine klare Zukunftsfähigkeit für PHP gegeben.	

Da PHP hier eher durch seine positiven Eigenschaften im Vergleich zu Perl hervorstach, wurde sich als Programmiersprache für PHP entschieden.

4.2 Abwägungen zwischen Client- und Serverlast

Bei der Konzipierung des Web Frontend ist es von entscheidender Bedeutung, wann und wofür JavaScript oder PHP eingesetzt wird. Auf der einen Seite ist nicht jeder Server sehr leistungsfähig, so dass eine interne Bildgenerierung eine große Auslastung verursachen könnte, welche man via JavaScript auf PC, Tablet oder Smartphone des Nutzers umleiten könnte. Auf der anderen Seite kann eine sehr intensive Bildgenerierung von mehreren Megabyte großen SVGs auch bereits ein Smartphone an seine Grenzen der Belastbarkeit führen.

Deshalb wurde mithilfe der folgenden Tabelle (vgl. **Tabelle 6: Vergleich zwischen Client- und Serverlast**) eine Abwägung zwischen Client-(JavaScript) und Serverlast (PHP) durchgeführt.

Tabelle 6: Vergleich zwischen Client- und Serverlast

Last auf Serverseite (PHP)	**Last auf Clientseite (JavaScript)**
PHP bietet mehr Sicherheit im Vergleich zu JavaScript.	Keine Garantie für korrekte Darstellung auf Nutzerseite.
Es ist einfacher / schneller von der Entwicklung her (Lesbarkeit und Einarbeitungszeit).	Leistung der Endgeräte (Smartphone,etc.) unbekannt.
Da Hausautomatisierung im ‚Alltag' jeweils nur eine begrenzte Anzahl an Personen (Eine Familie mit X Personen) dies nutzt, wird der Server nicht überlastet.	Server wird bei aufwändigerer Bildgenerierung entlastet.
Last auf Serverseite bei Steuerung des Hauses ist daher vernachlässigbar.	Wesentlich interaktiveres „Look and Feel" erreichbar, durch Bibliotheken wie z.B. JQuery (Animationen in der Oberfläche).
Aber: Last auf Serverseite bei komplexer Bildgenerierung nicht vernachlässigbar.	Automatisches Update von bestimmten Teilbereichen der Seite ohne Neuladen der gesamten Seite.

Nach Betrachtung dieser Grundgedanken wurde sich dafür entschieden, beide Technologien in Kombination zu verwenden, um zum einen die Vorteile von PHP und von JavaScript zu vereinen und so später durch Testen ein Optimum zu finden. Dabei kann auch in Zukunft über intelligentes Caching nachgedacht werden. Zusätzlich kann gesagt werden, dass mithilfe von Ajax und JavaScript eine deutlich dynamischere Seite erst überhaupt möglich wird.

4.3 Betrachtung verschiedener JS-Bibliotheken für das Charting

Da entschieden wurde, zusätzlich zu PHP auch JavaScript zu nutzen, bietet es sich an für die Bildgenerierung bereits bestehende Programmbibliotheken bzw. Frameworks zu finden und zu nutzen. Wichtige Kriterien waren neben den Interaktionsmöglichkeiten auch optische Aspekte, die Möglichkeit neben normalen Bildern auch Vektorgrafiken (SVG) zu erstellen und eine möglichst einfache Implementierung.

Dabei wurden Bluff, Canvas 3D Graph, CanvasXpress, ccchart, D3, dhtmlxChart, dygraphs, Flotr2, gRaphaël, Ico, JSXGraph, Google Chart, Protovis, Elycharts, Flot, jqPlot, MilkChart, PlotKit, FusionCharts, Highcharts und JS Charts betrachtet. Eine genaue Auflistung und Bewertung aller Tools ist im Anhang (siehe *Anlage 4: JavaScript Charting Bibliotheken*) zu finden.

Auswertend ist zu sagen, dass vor allem dygraphs, Flotr2 und Google Charts eine gute Grundlage für das Charting bieten würden.

dygraphs : http://dygraphs.com/

- Sehr ausgiebige Interaktion möglich
- Optisch überzeugend (professioneller Eindruck)
- Einfach zu implementieren

Flotr2: http://www.humblesoftware.com/flotr2/

- Recht gute Interaktion möglich
- Optisch überzeugend

Google Charts: https://developers.google.com/chart/interactive/docs/gallery?hl=de

- Keine Interaktion möglich
- Nur Vektorgrafiken
- Optisch in Ordnung

Die anderen Bibliotheken sind aufgrund von mangelhafter Optik, Interaktionsmöglichkeit, aufgrund von Lizensierungsproblemen oder aufgrund von einer zu aufwändigen Implementierung aus der Betrachtung herausgefallen. Die drei bereits genannten (siehe oben) Möglichkeiten zur Bildgenerierung werden daher im Forschungsseminar bevorzugt verwendet werden.

Zusätzlich soll erwähnt werden, dass es momentan bereits ein SVG-Modul in FHEM gibt, womit man eine grafische Auswertung von z.B. Sensordaten durchführen kann (vgl **Abbildung 2: SVG-Modul von Fhem zur Analyse**).

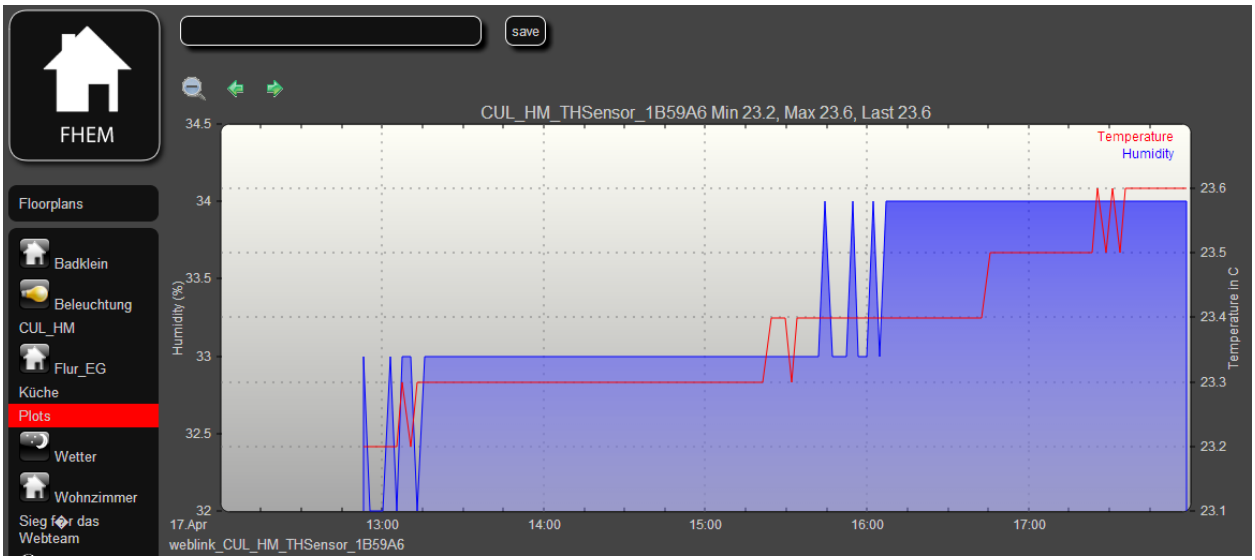

Abbildung 2: SVG-Modul von Fhem zur Analyse

Die Vorteile dieses Modules sind neben der guten Skalierbarkeit und Möglichkeit für grundlegende Interaktionen auch, dass es bereits in den FHEM-Server integriert ist. Nachteilig zu erwähnen wäre hierbei, dass die Konfiguration nur über aufwändiges Schreiben von Code möglich ist, die Interaktionen keine weitreichenderen Analysemöglichkeiten bieten und sich die Diagramme nicht automatisch aktualisieren (nur durch Neuladen der Seite).

Momentan ist ein sogenanntes ‚Neues Charting Frontend' (vgl. **Abbildung 3: Neues Charting Frontend für FHEM**) für den FHEM-Server in Entwicklung. Es ist noch recht neu, ein komplett eigenes Frontend und fällt damit aus unserer Betrachtung heraus. Trotz allem wirkt es sehr interaktiv und ist optisch ansprechend, aber da es recht neu ist gibt es noch keine Dokumentation oder Referenz dazu.

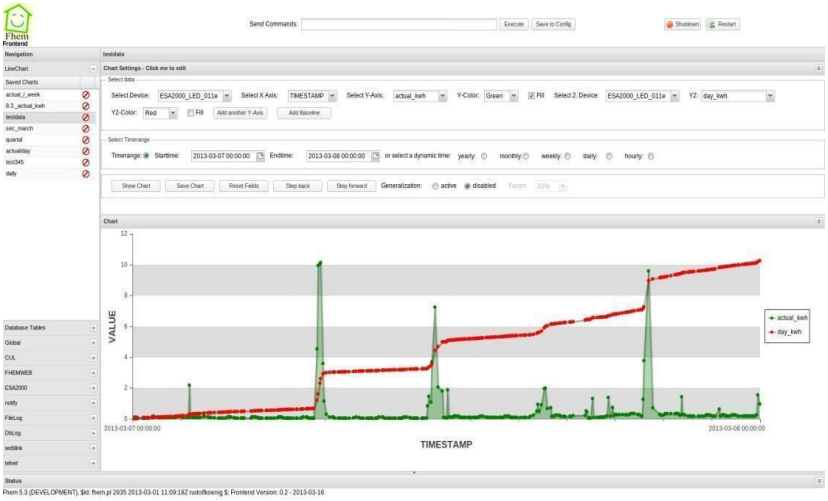

Abbildung 3: Neues Charting Frontend für FHEM[16]

[16] ([FHWI13b] Neues Charting Frontend aus der FHEMWiki, 2013)

4.4 Möglichkeiten zur Floorplanerstellung

Da ein Nutzer auch einen Floorplan mit angezeigten und steuerbaren Sensoren anlegen, nutzen und wieder löschen möchte, ist eine kurze Betrachtung verschiedener Möglichkeiten zur Generierung von Floorplans nötig. Dazu im Folgenden nun tabellarische Vergleiche und optische Eindrücke der vier verschiedenen, betrachteten Lösungen.

1. HTML basierte Lösung:

Die HTML basierte Lösung basiert auf einen Wiki-Eintrag[17] der FHEM Community und wird anhand der folgenden Tabelle (vgl. **Tabelle 7: Vor- und Nachteile der HTML basierten Lösung**) auf Vor- und Nachteile untersucht.

Tabelle 7: Vor- und Nachteile der HTML basierten Lösung

Vorteile	Nachteile
Bei Änderung des Systems lädt der Browser diese nach (über das Meta-Tag).	Vorgeschlagene Vorgehensweise entspricht nicht dem Stand der Technik. Es wird versucht AJAX Funktionalität zu erreichen ohne AJAX zu verwenden.
	Der Floorplan muss komplett selbst geschrieben werden, sehr zeitaufwändig und fehlerträchtig.

2. SVG basierte Lösung:

Die SVG basierte Lösung nutzt ein Unix-Tool[18], um einen Floorplan zu erstellen und in das Web zu integrieren. Anhand der folgenden Tabelle (vgl. **Tabelle 8: Vor- und Nachteile der SVG basierten Lösung**) wird diese Lösung auf Vor- und Nachteile untersucht.

Zusätzlich gibt es ein Beispiel (vgl. **Abbildung 4: SVG Beispiel**) als Bild zum besseren Verständnis der optisch mangelhaften Visualisierung.

[17] Vgl. hierzu ([FHWI13a] Grundriss mit Buttons aus der FHEMWiki, 2013).
[18] Vgl. hierzu ([SRCF13] fhzctrl von Sourceforge, 2013).

Tabelle 8: Vor- und Nachteile der SVG basierten Lösung

Vorteile	Nachteile
Es ist eine freie Skalierung des Floorplans möglich.	Diese Lösung befindet sich in einer Alpha Version.
	Ist seit 4-5 Jahren nicht mehr weiterentwickelt wurden.
	Optisch nicht sehr ansprechend.
	Konfiguration nicht richtig möglich.

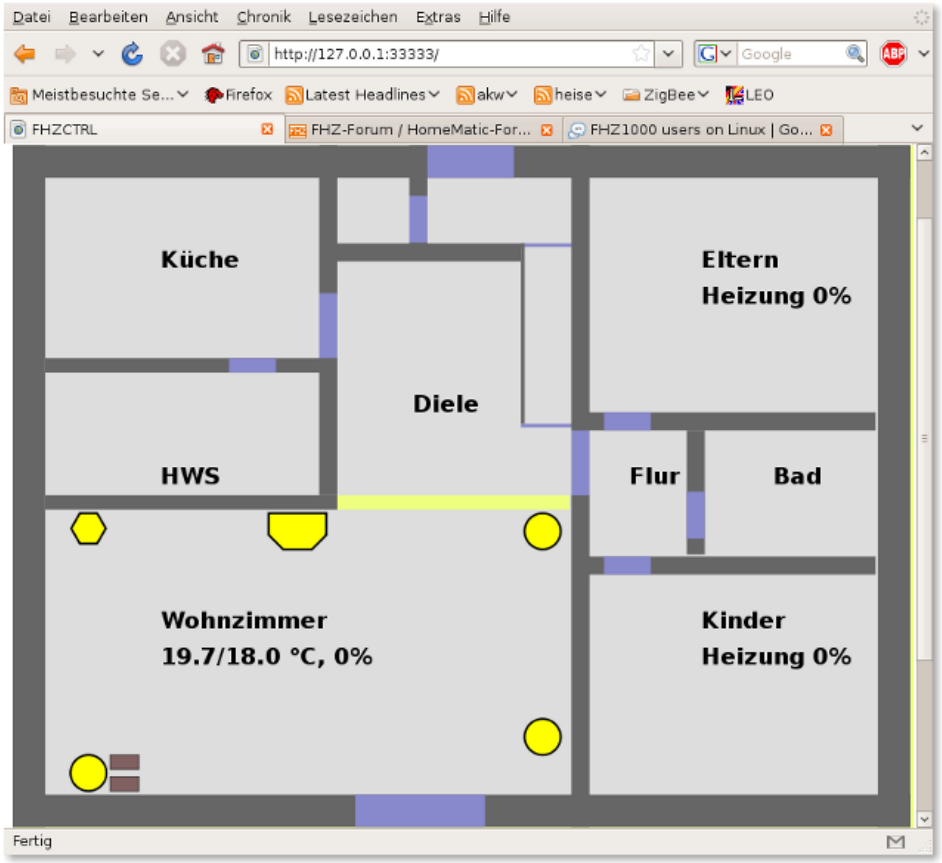

Abbildung 4: SVG Beispiel[19]

3. Lösung von HomeMini:

HomeMini[20] ist eine auf PHP- und Ajaxtechnologien basierende Floorplanlösung, die für FHEM eigens kreiert wurde. Anhand der folgenden Tabelle (vgl. **Tabelle 9: Vor- und**

[19] Vgl. hierzu ([SRCF13] fhzctrl von Sourceforge, 2013).
[20] In Anlehnung an ([HAWI13] HomeMini Beschreibung auf Hansemanns Wiki, 2013, 2012).

Nachteile von HomeMini) findet eine Betrachtung der Vor- und Nachteile dieser Lösung statt.

Tabelle 9: Vor- und Nachteile von HomeMini

Vorteile	Nachteile
Verwendet PHP- und AJAX-Technologien.	Verwendung von <area> Objekten für einen Vorhanden Floorplan. Floorplan und Devices bilden somit eine bildliche-Einheit. Probleme bei der Darstellung von Devices außerhalb des Bildes.
Kein Neuladen der kompletten Seite bei Änderung eines einzelnen Status .	Neugenerierung und Nachladen des Bildes bei jeder kleinsten Änderung
ToolTip zur Anzeige der Statusänderungsmöglichkeiten.	Konfiguration über manuelles editieren einer PHP-Datei unter Angabe der Koordinaten für die einzelnen Devices. Aufwendig und nicht nutzerfreundlich.

Als optisches Beispiel für HomeMini dient folgendes Bild (vgl. **Abbildung 5: HomeMini Beispiel**):

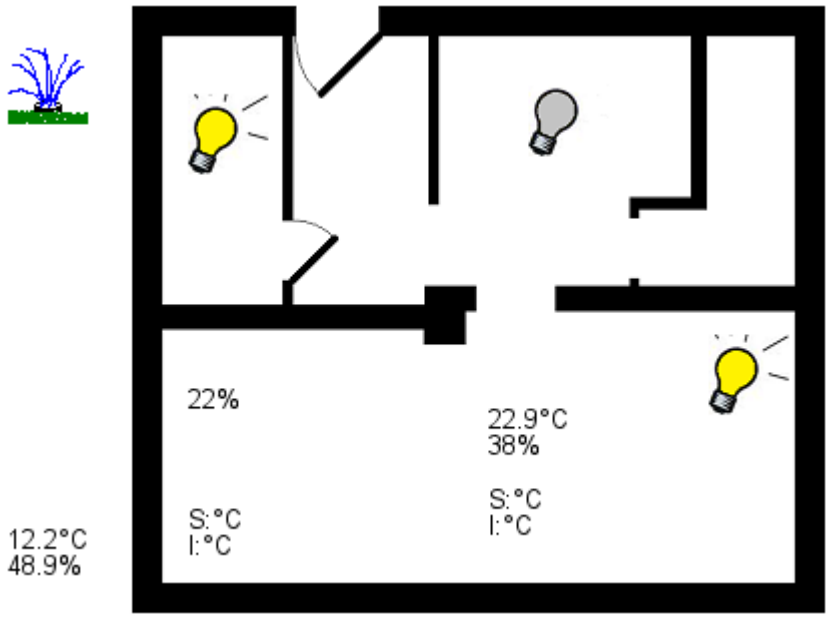

Abbildung 5: HomeMini Beispiel[21]

[21] Vgl. hierzu ([HAWI13] HomeMini Beschreibung auf Hansemanns Wiki, 2013, 2012).

4. YAF – Yet Another Floorplan

„***YAF*** *ist ein Floorplan, der es erlaubt, Geräte per Drag&Drop zu platzieren.*

(…) YAF entstand als Projektarbeit von Daniel Weisensee und Markus Mangei an der Hochschule Karlsruhe Technik und Wirtschaft.

Es steht für "Yet Another Floorplan" und soll eine Alternative zum bisher vorhandenen Floorplan bieten. YAF basiert auf Clientseite auf den JavaScript Frameworks JQuery und JQuery UI, (…) um die Konfiguration zu persistieren und um Daten zwischen der Oberfläche und dem Server austauschen wird die FHEM Konfiguration sowie eine Webschnittstelle verwendet.

Durch die Erweiterbarkeit von Widgets soll YAF flexibel gehalten werden. Mit Hilfe dieser Schnittstelle können problemlos Widgets von verschiedenen Entwicklern veröffentlicht werden, ohne dass sich diese über gewünschte Änderungen am YAF Code mit der Community abstimmen müssen. Es soll ähnlich dem Prinzip der Widgets unter Android oder Windows funktionieren. Widgets sollen speziell für FHEM Plugins geschrieben werden, um somit möglichst komfortable Oberflächen bieten zu können.

Das Projekt ist freie Software unter der GNU General Public License und befindet sich auf dem offiziellen FHEM SourceForge.net Repository: https://sourceforge.net/p/fhem/code/HEAD/tree/trunk/fhem/contrib/YAF/ "[22]

Die folgende Abbildung (vgl. **Abbildung 6: Yet Another Floorplan**) zeigt ein optisches Beispiel von YAF. Aufgrund der begrenzten Zeit des Forschungsseminares ist noch keine intensive Abwägung und Untersuchung auf Vor- und Nachteile von YAF vorhanden.

[22] Vgl. hierzu ([WEMA13] Daniel Weisensee Markus Mangei YAF-Projekt, 2013)

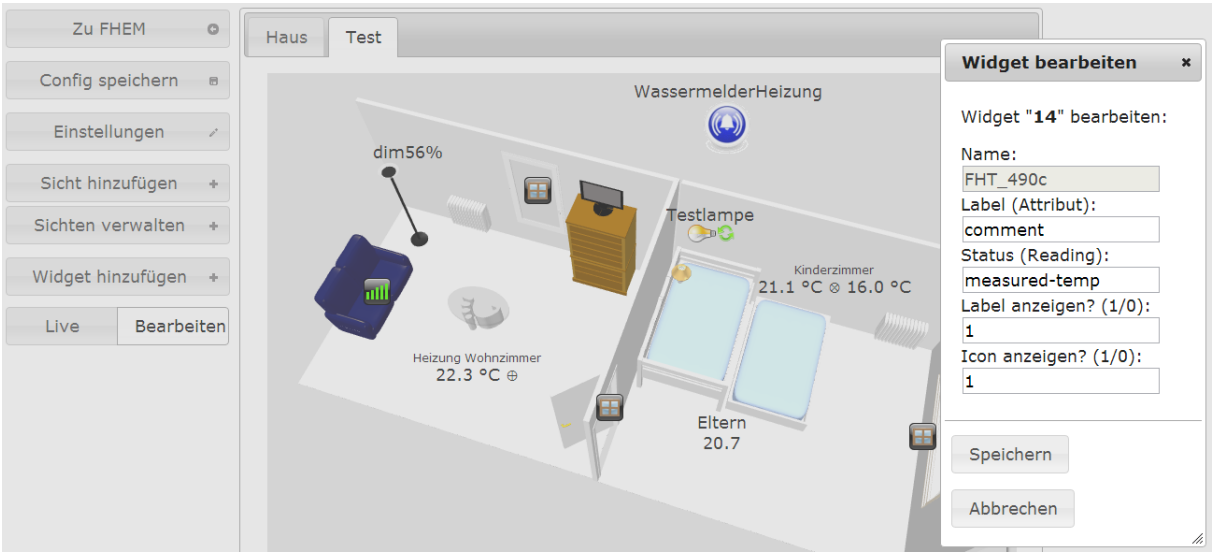

Abbildung 6: Yet Another Floorplan[23]

5. Das in FHEM integrierte Floorplan-Modul:

Das bereits in FHEM integrierte Floorplan-Modul ist standardmäßig bei der Installation mit vorgegeben. Eine Abwägung der Vor- und Nachteile findet in der folgenden Tabelle (vgl. **Tabelle 10: Vor- und Nachteile des in FHEM integrierten Floorplan-Moduls**) statt:

Tabelle 10: Vor- und Nachteile des in FHEM integrierten Floorplan-Moduls

Vorteile	Nachteile
Schon direkt im FHEM eingebunden. Sehr leichte Konfiguration (ohne Coding), mit großen Einstellungsmöglichkeiten.	Bei der Änderung eines Devices-Status wird die gesamte Seite neugeladen.
Der Floorplan fügt sich gut in die bisherige FHEM Umgebung ein.	Durch die Vererbung des FHEMWEB „attr refresh“ sollte eigentlich die Web Page nach den angegeben Sek. neu geladen werden. Dies funktionierte im Test aber nicht. (basiert auf einem <meta> Tag)
Devices können auch außerhalb des Floorplans „modelliert“ werden.	Technologisch gesehen ausbaufähig (kaum Einsatz von JavaScript)
Jede erdenkliche Art des Ausbaus über das Einfügen von HTML-Code möglich	
Freie Layout-Konfiguration über eigene Floorplan .css Datei	

[23] Vgl. hierzu ([YAF13] Screenshot FhemWiki zu YAF, 2013)

Zusätzlich dient als optisches Beispiel folgende Abbildung (vgl. **Abbildung 7: Integrierter FHEM-Floorplan Beispiel**):

Abbildung 7: Integrierter FHEM-Floorplan Beispiel

Zusammenfassend muss gesagt werden, dass das in FHEM integrierte Floorplan-Modul momentan am besten in der Betrachtung abschneidet, jedoch auch hier deutliche Verbesserungen nötig sind. Eine Entwicklung eines eigenen Floorplan-Modules ist allerdings nicht Teil dieses Forschungsberichtes und ein Anwendungsgebiet für die Zukunft.

4.5 Bereits vorhandene Software-Lösungen auf dem Markt

Zur Orientierung und für die Erstellung eines erfolgreichen Konzeptes ist es sinnvoll, die Lösungen bereits am Markt etablierter Firmen zu betrachten. Da diese meist proprietärer Natur sind und Kostenmodelle für die langfristige Nutzung durch Kunden etabliert haben, soll sich in der Gruppe durch die reine Betrachtung und Benutzung von Testversionen ein besseres Grundverständnis zum Umgang mit Web Frontend angeeignet werden.

Dabei wurden die Produkte von IP-Symcon, Homematic, Homeputer, EZcontrol, Marmitek und RWE Smarthome im Rahmen der erreichbaren Möglichkeiten betrachtet. Im Folgenden eine Betrachtung der Lösungen, welche rundum zufriedenstellend waren und einen professionellen Eindruck vermittelten.

IP-Symcon:

- Ein Test des Web Frontend ist möglich über: http://www.webfront.info/
- Die Erstellung fand mithilfe der Google API's statt.
- Durch starke Nutzung von JavaScript ist eine hohe Interaktivität mit dem System möglich.
- Die Last liegt auf Clientseite, da die Seitengenerierung im JavaScript stattfindet.
- Die Seite zeichnet sich durch eine starke Nutzung von Ajax aus.
- Miteinander abgestimmte Farbverläufe sorgen für ein optisch gutes Nutzererlebnis.
- Die Menüs sind hierarchisch gegliedert.
- Die Bedienung ist selbsterklärend und intuitiv.
- Die Interaktivität mit dem System ist an mehreren Stellen geben, insgesamt jedoch eher gering.
- Es findet eine klare Trennung der Bedienung vom der Konfiguration statt.

RWE Smarthome:

- Ein Beispielvideo zur Nutzung von RWE Smarthome ist unter folgendem Link zu finden: http://www.youtube.com/watch?v=84RRXyy-AkE
- Diese Lösung wurde in Silverlight geschrieben.
- Die Steuerung basiert auf Drag & Drop Mechaniken und ist sehr intuitiv.
- Die Navigation basiert auf einem komplett eigenen Konzept, was jedoch leicht verständlich ist.
- Diese Lösung ist sehr interaktiv und ansprechend.

Die Lösungen der anderen Firmen sind aufgrund schlechter Optik, unverständlicher Menüführung oder mangelnder Informationen über weitergehende Konzepte aus der Betrachtung hinaus gefallen. Weitere Informationen zu den anderen Produkten sind im Anhang (siehe *Anlage 5: Vorhandene Software-Lösungen des Marktes*) zu finden.

4.6 Sonstige Betrachtungen

Es ist weiterhin möglich, über eine Java-Schnittstelle mit dem FHEM-Server zu kommunizieren. Dazu wurde im Rahmen einer Diplomarbeit der Universität Karlsruhe[24] ein Projekt gestartet. Diese Anbindung ist für das Forschungsseminar im Rahmen der Entwicklung eines Web Frontend nicht relevant, kann aber für zukünftige Betrachtungen (Arbeit mit Servlets und Web Services) weiter untersucht werden.

[24] Vgl. hierzu ([GEKI13] FHEM JAVA-Schnittstelle, 2013)

5. Entwurf

Basierend auf dem angeeigneten Wissen über dem im Forschungsseminar verwendeten FHEM-Server (siehe *2. Kurzbetrachtung des FHEM-Server*), der umfassenden und abstrahierten Anforderungsanalyse (siehe *3. Anforderungsanalyse*) und der technologischen Betrachtungen (siehe *4. Technologische Betrachtungen*) wurde nun ein Konzept zum allgemein Aufbau des Web Frontend erstellt und wird in diesem Kapitel vorgestellt. Dieses Kapitel soll in Zukunft von weiteren Gruppen oder Interessierten erweitert werden.

5.1 Verwendete Technologien und Kompatibilität

Das Ziel ist, eine Kompatibilität für Firefox (ab Version 26), Chrome (ab Version 31), IE (ab Version 11), Smartphone (ab 720p), Tablet und PC zu erreichen. Zeitgleich soll das Web Frontend unabhängig vom Inhalt des Datenmodells des FHEM-Servers funktionieren können.

Aufgrund von Sicherheitsalgorithmen wird eine PHP-Version ab 5.3.7 benötigt. Ab Version 5.5 sind die sicheren Algorithmen zur Hashgenerierung (mit Salt) bei Passwörtern bereits direkt in PHP integriert, von Version 5.3.7 bis Version 5.4.9 wird eine PHP-Zusatzbibliothek verwendet. Zeitglich ist diese Mindestversion auch wegen der verwendeten SQLite-Datenbank erforderlich.

Das Web Frontend generiert HTML5-Dokumente und das Layout wird mit CSS3-Elementen gesteuert. Zusätzlich wird mithilfe von JavaScript clientseitig die Steuerung des Views übernommen.

5.2 Infrastruktur

Aufgrund der gewünschten Kompatibilität (siehe *5.1 Verwendete Technologien und Kompatibilität)* ist es notwendig, eine geeignete Infrastruktur zu finden, welche es ermöglicht, den View und die Models auszutauschen. Dazu fand eine Anlehnung an das MVC-Muster statt (vgl. **Abbildung 8: Infrastruktur des Web Frontend**).

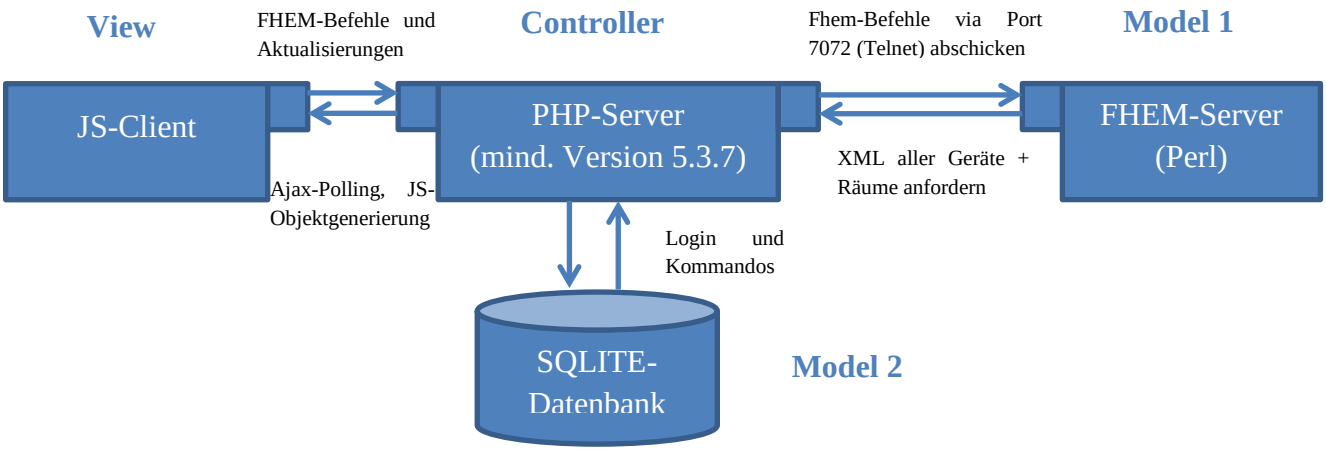

Abbildung 8: Infrastruktur des Web Frontend

Der FHEM-Server dient in der aufgezeigten Infrastruktur als Datengrundlage. Es ist möglich, via Telnet sämtliche, benötigte Daten über XML anzufordern. Da das XML lediglich lesend benötigt wird, wird ein SAX-Parser aufgrund der erhöhten Geschwindigkeit verwendet.

Der Controller, welcher den SAX-Parser beinhaltet, wird nach Verbindungsaufbau mit dem FHEM-Server das XML anfordern und die Daten zu PHP-Objekten mit zugehörigen Klassen transformieren. Der Controller erstellt ebenso eine Verbindung zu einer SQLITE-Datenbank, in der Nutzer unter Anderem mit Passwörtern und Rechten verwaltet werden.

Zur Performancesteigerung werden sämtliche PHP-Objekte zeitgleich auch als JavaScript-Objekte redundant gespeichert und aktuell gehalten. Dies ist eine Form des Caching auf Nutzerseite und erhöht die Performance.

Änderungen, die sich aus der Interaktion mit dem View ergeben, werden nach Prüfung mit Befehlen an den FHEM-Server geschickt, so dass Daten geändert, gelöscht oder hinzugefügt werden können (vergleichbar mit dem Abschicken von SQL-Befehlen an eine Datenbank, um das Model zu ändern).

Der JS-Client ist dafür zuständig, den View zu generieren. Wie oben erwähnt, fand dies als eine Anlehnung an ein MVC-Muster statt. Aufgrund der Natur mancher Internettechnologien (primär Ajax) ist eine komplette Trennung zwischen Controller und View nicht möglich. Es wurde beim Entwurf darauf geachtet, die Überschneidungen so gering wie möglich zu halten. Von daher werden fast alle Aufgaben, die von einem Controller zu erwarten sind, via Ajax als Aktualisierungen und Befehle an den Controller auf Serverseite geschickt. Dies wird zeitgleich dazu genutzt, eine client- und serverseitige Überprüfung der Eingabe und Daten zu vollziehen.

5.2 Datenmodelle

Da die Daten für die Hausautomatisierung vom FHEM-Server kommen, muss eine Kurzerklärung des Aufbaus der Daten stattfinden. Zeitgleich findet eine Betrachtung der Datenstruktur in der verwendeten SQLite-Datenbank statt. Das Datenmodell ist künftig erweiterbar und nicht vollständig.

5.2.1 FHEM-Server

Die benötigten Daten des FHEM-Servers lassen sich in zwei Oberkategorien einteilen: Geräte und Räume. Geräte können Aktoren oder Sensoren sein und enthalten spezifische Informationen über ihre Zustände, Zustandswechsel, Typ, Modell, Aliasnamen und Messwerte. Räume sind in FHEM Attribute von Geräten und im Web Frontend Klassen, in denen kein, eines oder mehrere Gerät(e) zugeordnet sein können. Ein Spezialfall der Räume ist ein ‚Structure', mit dem z.B. alle Geräte eines Typs zusammengefasst werden können.

Die Abbildung der Geräte und Räume wird über Klassen (abstrakte und nicht abstrakte) erfolgen.

5.2.2 SQLite-Datenbank

Die SQLite-Datenbank enthält momentan zwei Tabellen. Eine ist für die Nutzersteuerung zuständig (vgl. **Tabelle 11: Table 'users'**).

Tabelle 11: Table 'users'

Spaltenname	Datentyp
user_id	integer (primary key)
user_name	varchar(64)
user_password_hash	varchar(255)
user_email	varchar(64)
user_role	varchar(64)

Zusätzlich wurde auf der Spalte ‚user_name' ein Unique Index (ASC) mit Namen ‚user_name_UNIQUE' erstellt.

Als Rollen gibt es momentan ‚User' und ‚Administrator'. Der User darf alle Geräte und Räume sehen, sowie mit Sensoren und Aktoren interagieren. Sämtliche weitere Veränderungen und Steuerungsmöglichkeiten sind dem Administrator vorbehalten.

Die zweite Tabelle enthält sämtliche vorhandenen Befehle des FHEM-Servers. Dies ist eine virtuelle Tabelle, welche mit FTS4 arbeitet (vgl. **Tabelle 12: Virtual Table ‚commands' (mit FTS4-Datentypen).**

Tabelle 12: Virtual Table ‚commands' (mit FTS4-Datentypen)

Spaltenname	Datentyp
id	id
command_name	text

Die Erstellung des Primärschlüssels auf der Spalte ‚id' und die Indexierung auf ‚command_name' findet bei einer virtuellen FTS4-Tabelle automatisch statt. Diese Tabelle wird für ein automatisches Vorschlagen von FHEM-Befehlen in der Kommandozeile benötigt. Neu hinzukommende Befehle für den FHEM-Server sollten in der Datenbank eingepflegt werden, damit die Suchvorschläge optimal sind.

5.3 Steuerung und Anordnung des Web Frontend

Das Web Frontend wird aus drei statischen Leisten bestehen, wobei die obere Leiste einen Button für Kommandozeile, Floorplan, Home, Steuerung und A/E-Mode (Anfänger/Experten-Modus, nur als Administrator sichtbar) hat (vgl. **Abbildung 9: Anordnung des Home-Bereich des Web Frontend)**.

Der Bereich ‚Home' soll dabei den Nutzer vielerlei Funktionalitäten auf der ersten Ebene und dem ersten Blick bieten. So werden Kacheln für jeden angelegten Raum kreiert, in dem der Nutzer zum einen eine Übersicht über alle Geräte des Raumes erhält. Die Kacheln sollen in der Größe je nach Anzahl der Geräte variieren, um bei der Anzeige auf dem Smartphone und Tablet Platz zu sparen.

Der Teilbereich links der Website wird dem An- und Abmelden dienen und die Möglichkeit für nutzerspezifische Einstellungen (Erstellung eigener Themes, Änderung des Passwortes, Setzen von Rechten, etc.) eröffnen. Im Rahmen der Erweiterbarkeit werden hier zusätzliche

Funktionalitäten eingebaut (vgl. **Abbildung 9: Anordnung des Home-Bereich des Web Frontend)**.

Abbildung 9: Anordnung des Home-Bereich des Web Frontend

Die obere und die linke Leiste werden mithilfe eines Pfeilbuttons ein- und ausklappbar sein. Eine dritte Leiste wird dem Administrator beim Anordnen von Räumen zu Geräten sichtbar sein (Button Gerätezuordnung). Diese Leiste wird sich rechts befinden und die nicht zugeordneten Geräte sortiert nach Modell geben. Beim Scrollen soll sich die rechte Seite mitbewegen und eine eigene Scroll-Leiste haben. Das Verschieben von Geräten zwischen den Räumen soll via Drag&Drop möglich sein.

Im Modus der Gerätezuordnung sollen Räume mithilfe eines ‚+'-Buttons hinzugefügt werden und direkt über die Tastatureingabe benannt werden können. Ein ‚-'-Button an jedem Raum ermöglicht das Löschen des Raumes und das Rücksetzen der Geräte auf den Raum ‚nicht zugeordnet'.

Im Modus der Benutzersteuerung (Button Benutzersteuerung) soll es dem Administrator möglich sein, Nutzer hinzuzufügen, zu löschen, das Passwort (unter Eingabe des alten Passworts) und die E-Mail zu ändern, sowie sich eine Nutzerübersicht anzuzeigen. Da die

Nutzung des Web Frontend primär für den privaten Bereich mit Familien konzipiert ist, ist dies ausreichend.

Mit dem Button ‚Kommandozeile' soll eine Kommandozeile ausgefahren werden, die via Vorschläge nach dem Eingeben der ersten Buchstaben Vorschläge für den gesuchten Befehl gibt (siehe *5.2.2 SQLite-Datenbank*). Beim Ausfahren der Kommandozeile soll die obere Leiste aus Platzgründen eingefahren werden.

Zusätzlich wird es die Möglichkeit geben, zwischen einem Anfänger- und einem Expertenmodus (A/E-Mode (A)) umzuschalten, wobei der Anfängermodus eher den Wert auf eine grafische Nutzung des Web Frontend setzt, der Expertenmodus zusätzlich weitere Eigenschaften und die wahren Namen der Geräte (vgl. **Abbildung 10: A/E-Modus mit Dropdown,** die Geräte haben alle einen Aliasnamen) anzeigt, so dass man auch effektiv mit der Kommandozeile arbeiten kann.

Abbildung 10: A/E-Modus mit Dropdown

Die untere, rechte Ecke eines jeden Gerätes soll zusätzlich für den Administrator im Expertenmodus als Pfeil anklickbar sein. Beim Klick auf die eingefärbte Ecke sollen sämtliche Zusatzinformationen des Gerätes, welche für Entwickler interessant sind, angezeigt werden (vgl. **Abbildung 10: A/E-Modus mit Dropdown)**.

Die gesamte Website soll sich dynamisch auf die Fenstergröße (und damit der Anzeige der Räume pro Zeile) anpassen können, um Platz zu sparen.

Das Klicken auf einen Aktor (z.B. Lampe) soll eine Aktion auslösen. Dies soll in Abhängigkeit des Typs des Gerätes variieren. So wird eine normale Lampe mit einem Klick aktiviert bzw. deaktiviert werden können, eine Dimmlampe wird hingegen mit einem Slider gesteuert. Diese Unterscheidung ist anhand der gegebenen Daten des FHEM-Servers (siehe *5.2.1 FHEM-Server*) über Klassen und Typen möglich. Bei einem Klick auf die Sensoren, soll eine grafische Analyse der Daten möglich werden (vgl. **Abbildung 11: Analyse von Sensorwerten**).

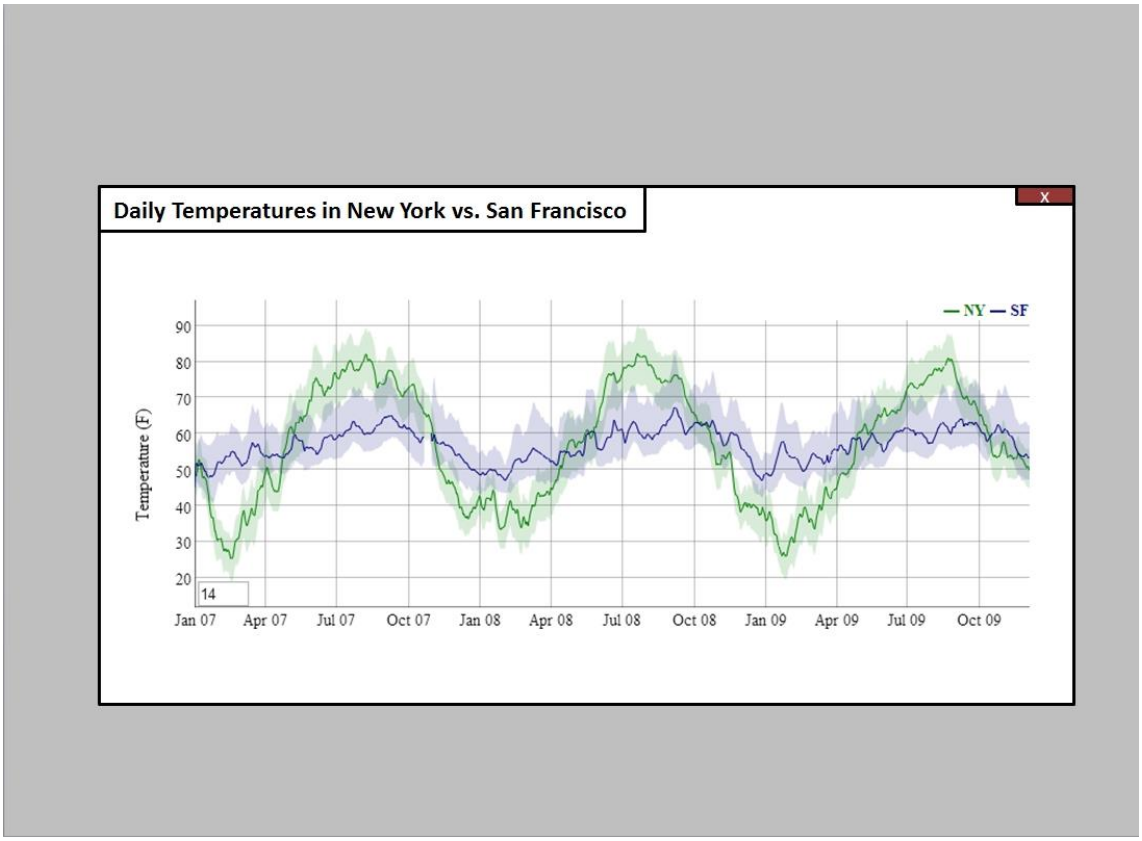

Abbildung 11: Analyse von Sensorwerten

Beim Klick auf ‚Floorplan' soll die Erstellung und Steuerung von Floorplans möglich sein. Dies ist in Zukunft zu implementieren und nicht Teil des Forschungsseminares im WS 2013/14.

Der Bereich Steuerung wird nur als Administrator sichtbar sein. Die gesamte Regelverarbeitung soll über diesen Bereich gesteuert werden. Die Implementierung der Regelverarbeitung ist nicht Bestandteil des Forschungsseminars im WS 2013/14.

6. Prototypische Implementierung

Auf Basis des Soll-Konzeptes (siehe *5. Entwurf*) erfolgt nun eine prototypische Implementierung des Web Frontend. Aufgrund der zeitlichen Begrenzung des Forschungsseminares und der Komplexität der Anwendungsfälle konnten nicht alle Anforderungen umgesetzt werden. Der Fokus lag dabei auf den Anwendungsfällen, welche mit Priorität 1 markiert wurden (siehe *Anlage 2: Allgemeine, funktionale Anforderungen*, **Abbildung 16: Interaktion des Administrators mit dem System**).

Für die prototypische Implementierung wurden ebenso eigene Bilder für Slider, Rollos, Lampen, etc. erstellt. Diese sind für die FHEM-Community, weiteren Interessierten oder Nachfolger im Forschungsseminar im Rahmen der Erstellung oder Erweiterung eines Web Frontend frei verfügbar.

In der im Entwurf vorgestellten Infrastrukturskizze (siehe *5.2 Infrastruktur*, **Abbildung 8: Infrastruktur des Web Frontend**) wurde bis auf das automatisierte Ajax-Polling zur Überprüfung der Veränderung der Daten alles als Grundlage implementiert. Des Weiteren wurden bis auf die Regelverarbeitung sämtliche Anforderungen mit Priorität 1 (siehe oben) implementiert. Zukünftige Interessierte oder Nachfolger sollten sich daher zuerst auf das Ajax-Polling und die Implementierung der Regelverarbeitung fokussieren, ehe sie Funktionalitäten mit niedrigerer Priorität einbauen.

6.1 Bisher genutzte Frameworks und Tutorials

Da es vor allem im Bereich der Sicherheit von Passwörtern immer sinnvoll ist, ein bekanntes und öffentliche Hashverfahren zu verwenden, anstatt selber einen Algorithmus zu implementieren und ggf. dabei einen Fehler einzubauen, wurde die gesamte Hashgenerierung (mit Salt) für die Passwörter, ebenso wie das grundlegende Gerüst der Login Prüfung des Controllers von PHPLogin[25] übernommen. Für den automatischen Vorschlag von Befehlen für die Kommandozeile wurde ein Tutorial von Papermashup[26] verwendet und erweitert.

[25] Vgl. hierzu ([PHPL13] PHPLogin.net, 2013)
[26] Vgl. hierzu ([PAPE13] JQuery-PHP-Ajax-Autosuggest, 2009)

Für die Nutzung und Erstellung der HTML5-Dokumente und Nutzen des CSS3 sowie JavaScript wurden Anleitungen der W3CSchool[27] verwendet (primär für Drag&Drop) und eigens erweitert.

Zusätzlich wurden JQuery, JQueryUI und JQuerySlimScroll als Frameworks für JavaScript eingebunden und verwendet.

6.2 Aufbau des Web Frontend

Der Aufbau des Web Frontends entspricht bis auf das automatische Ajax-Polling der Infrastruktur des Entwurfes (siehe *5.2 Infrastruktur*). Im Folgenden findet eine Betrachtung der Dateien und Verzeichnisse, sowie der Klassen und Funktionen des Projektes statt.

6.2.1 Dateien und Verzeichnisse

Das Projekt enthält ein Verzeichnis mit dem Namen **‚database‘**. Darin enthalten ist die SQLite-Datenbankdatei ‚fhem.db‘, in der die Inhalte der Datenbank gespeichert sind. Zusätzlich befindet sich eine .htacess Datei mit dem Inhalt ‚deny from all‘ im Verzeichnis, um Zugriffe von außerhalb zu verhindern.

Das Projekt enthält weiterhin ein Verzeichnis mit dem Namen **‚img‘**. Darin enthalten sind sämtliche im Projekt verwendeten Bilder und Dateien.

Das dritte Verzeichnis des Projektes heißt **‚libraries‘**. Darin enthalten sind sämtliche für das Projekt verwendeten Bibliotheken. Die **‚password_compatibility_library.php‘** wird nur dann geladen, wenn die installierte PHP Version zwischen 5.3.7 und 5.4.9 liegt, die JavaScript-Frameworks (siehe 6.1 *Bisher genutzte Frameworks und Tutorials*) werden immer direkt in das Projekt eingefügt.

[27] Vgl. hierzu ([W3CS13] W3CSchools, 2013)

Die folgenden Dateien des Projektes werden nun tabellarisch (vgl. **Tabelle 13: Dateien des Web Frontends**) samt ihrer Aufgaben aufgezeigt.

Tabelle 13: Dateien des Web Frontends

Dateiname	Aufgabe
_install.php	Erstellung der Datenbank fhem.db mit Administrator ‚Admin' und Standardpasswort ‚Test123'.
Auslesen.php	Auslesen des XML und Anlegen von PHP-, sowie JS-Objekten für Räume und Klassen nach vordefinierter Klassenstruktur. Diese Datei ist ebenso für das Generieren von Menüzeilen zuständig. Dies geschieht in Abhängigkeit der Rechte des eingeloggten Nutzers. Zusätzlich werden hier die unterstützten Models definiert.
autosuggest.php	Zuständig für die Bereitstellung der Suchvorschläge für die Kommandozeile. Dabei findet eine Suche in der Datenbank statt.
FHEM_Anfrage.php	Zuständig für die Übermittlung von Befehlen an den FHEM-Server. Kann im ‚save'-Modus aufgerufen werden, so dass die Befehle in der FHEM.cfg gespeichert werden (je nachdem ob man Entwickler oder Nutzer ist).
index.php	Standardseite und Controller der Anwendung. Erstellt Verbindung mit FHEM-Server und Datenbank, kontrolliert das Login und die Session. Lädt benötigte Dateien aus dem ‚libraries'-Ordner nach.
scripts.js	JavaScript, wenn man noch nicht angemeldet ist.
scripts_admin.js	Bereitstellung von Autosuggest(Kommandozeile) für Admin, Anlegen von Events für das Verschieben von Objekten (Drag&Drop), sowie Anlegen von neuen Räumen und Gerätezuordnung im Vergleich zu normalem ‚User'. Die Benutzersteuerung und der A/E-Modus werden hier ebenso gesteuert.
scripts_nachher.js	Langsames Fading der Räume sowie Änderung der Objekte vom display-type:none auf korrekte Anzeige (da sonst alle Räume mit Geräten in der oberen, linken Ecke wären)
scripts_user.js	Nutzung der Klassen- und Objektstruktur der Auslesen.php-Datei, um Geräte und Räume im JS zu halten. Hier wird die Darstellung der Räume erzeugt.
style.css	CSS3-Layout für die Website.
user_control.php	Zuständig für die Benutzersteuerung (Nutzer anzeigen, anlegen, ändern, löschen) für den Admin

6.2.2 Klassen

Folgende, grundlegende Klassen werden im Web Frontend verwendet:

abstract class Device:

Klasse mit eigenen Konstruktor, get()- und set()-Methoden, sowie protected Variablen, die für alle Geräte zutreffend sind. Die abstrakte Klasse Device wird durch Spezialisierung von mehreren, weiteren Klassen genutzt.

class Device_FS20 extends Device:

Diese Klasse ist für alle Geräte vom Typ ‘FS20’ zuständig. Neben dem Konstruktor, sowie get()- und set()-Methoden und protected Variablen ist diese Klasse auch in der Lage, das Device als HTML und JS-Objekt auszugeben (bzw. zu generieren).

class Device_FHT extends Device:

Diese Klasse ist für alle Geräte vom Typ ‘FHT’ (Heizung) zuständig. Neben dem Konstruktor, sowie get()- und set()-Methoden und protected Variablen ist diese Klasse auch in der Lage, das Device als HTML und JS-Objekt auszugeben (bzw. zu generieren).

class Device_HM extends Device:

Diese Klasse ist für alle Geräte vom Typ ‘HM’ (Homematic) zuständig. Neben dem Konstruktor, sowie get()- und set()-Methoden und protected Variablen ist diese Klasse auch in der Lage, das Device als HTML und JS-Objekt auszugeben (bzw. zu generieren).

class Device_WSN extends Device:

Diese Klasse ist für alle Geräte vom Typ ‘WSN’ (Gerätetyp des Forschungsseminares) zuständig. Neben dem Konstruktor, sowie get()- und set()-Methoden und protected Variablen ist diese Klasse auch in der Lage, das Device als HTML und JS-Objekt auszugeben (bzw. zu generieren).

class Room:

Diese Klasse ist für alle Räume zuständig. Neben dem Konstruktor, sowie get()- und set()-Methoden und protected Variablen ist diese Klasse auch in der Lage, Geräte dem Raum zuzuordnen.

class FHEM_XML_Parser:

Diese Klasse ist für den SAX-Parser zuständig. Neben dem Konstruktor, sowie get()- und set()-Methoden und protected Variablen ist diese Klasse auch in der Lage, den SAX-Parser zum Auslesen des XML zu verwenden.

6.3 Interne (funktionale) Abläufe

Nach der Betrachtung der Dateien und Verzeichnisse (siehe *6.2.1 Dateien und Verzeichnisse*) und Klassen (siehe *6.2.2 Klassen*) findet nun eine Betrachtung der grundlegenden Funktionen statt, welche im Projekt verwendet werden.

Zusammenfassend verfügt das Web Frontend über folgende Funktionalitäten:

- funktionierendes Loginsystem mit Hash(+Salt)-Generierung via SQLite-Datenbank
- funktionierende Kommandozeile mit Suchvorschlägen
- funktionierender SAX-Parser zum Auslesen des XML des FHEM-Server
- funktionierende Generierung von PHP-Objekten (Serverseite) und JS-Objekten (Clientseite) zum Performancegewinn
- funktionierender Algorithmus für korrekte Darstellung der Geräte und Räume in IE, Firefox und Chrome, sowie richtiges Generieren/Anpassen der Website bei Verschiebung von Objekten und Änderung der Fenstergröße auf PC, Tablet und Smartphone
- funktionierendes Drag&Drop für intuitives Verschieben von Aktoren und Sensoren in Räumen
- Rückkopplung der Interaktionen des Clients (JS) an den Controller (PHP)
- Anlegen neuer Räume, Namensvergabe und Möglichkeit, Aktoren und Sensoren dort hineinzuziehen, löschen der Räume
- Nachziehen der wahren Namen im A/E-Modus
- funktionierende Benutzersteuerung und Gerätezuordnung
- funktionierende Gerätesteuerung und Anzeigen aller Attribute im A/E-Modus
- funktionierendes Erkennen des Geräts der Forschungsgruppe

6.3.1 Erstellung der SQLite-Datenbank

Mithilfe der ‚_install.php' wird im Ordner ‚database' eine Datei ‚fhem.db' erzeugt. Dabei wird eine SQLite-Datenbank erstellt und mit den benötigten Tabellen (siehe *5.2.2 SQLite-Datenbank*) gefüllt. In Abhängigkeit der installierten PHP-Version (mindestens 5.3.7) wird ggf. die ‚password_compatibility_library.php' aus dem Ordner ‚libraries' hinzugefügt.

Der Standardnutzer ‚Admin' (Administratorrechte) mit dem Passwort ‚Test123' wird dabei angelegt. Das Passwort sollte vom Nutzer so früh wie möglich über die Benutzersteuerung des Web Frontend geändert werden.

Nach der Installation sollte die ‚_install.php' in den Ordner ‚database' verschoben werden, um einen Zugriff oder Angriff von außen zu verhindern.

6.3.2 Initialisierung und Generierung der Anzeige

Zu Beginn wird in der ‚index.php' die Verbindung zur SQLite-Datenbank aufgebaut und ggf. je nach PHP-Version Bibliotheken nachgeladen. Die Sessionverwaltung wird gestartet und eine Verbindung zum FHEM-Server aufgebaut. Die Login-Funktionalität wird bereitgestellt. Nach dem Login wird der XML-Parser initialisiert und das XML von FHEM angefordert. Zusätzlich wird ein Array mit allen vorhandenen Räumen generiert, was den Übergang zur ‚Auslesen.php' darstellt.

In der ‚Auslesen.php' das gesamte XML bezüglich der Geräte und Räume ausgewertet, sowie PHP- und JavaScript-Objekte anhand der vordefinierten Klassen- und Objektstruktur erzeugt. Die Menüzeilen werden in Abhängigkeit der Rolle des eingeloggten Users generiert. Die verwendeten JS-Objekte werden in der ‚scripts_user.js' und ‚scripts_admin.js' verwendet.

In der ‚scripts_user.js' wird mithilfe der JS-Objekte die Funktion ‚Room_Renderer(breite)' als grundlegende Funktionalität zur korrekten Darstellung aller Räume und ihrer Geräte aufgerufen. Sie wird ebenso aufgerufen, wenn es eine Nutzerinteraktion (Fenstergröße verändert, verschieben von Objekten) stattfand.

Zusätzlich wird in der script_user.js die Funktionalität für das Klicken der Geräte und Sensoren bereitgestellt (als Eventhandling).

Sollte der Nutzer als Admin eingeloggt sein, wird die ‚scripts_admin.js' mit eingebunden. Sie stellt die Kommandozeile mit Suchvorschlägen bereit und erstellt Events für das Verschieben von Objekten (Drag&Drop), Anlegen/Ändern von Räumen und Gerätezuordnung bereit. Die Menüführung unterscheidet sich in diesen Punkten vom normalen User.

In der ‚scripts_nachher.js' wird das langsame Fading der Räume beim ersten Öffnen der index.php bereitgestellt. Die Objekte werden in dieser Datei als vorletzter Schritt vom display-type:none auf die korrekte Anzeige geändert. Andernfalls wären alle Räume oben links in der Ecke übereinander gestapelt.

Nach der Bereitstellung aller JS-Funktionalitäten wird in der ‚Auslesen.php' das zu verwendende HTML und JS kombiniert und an die ‚index.php' weitergegeben.

Die ‚index.php' stellt nun das generierte HTML und JS bereit.

6.3.3 Erweiterte Steuerungsmöglichkeiten (Administrator)

Wenn ein Nutzer mit Administratorrechten eingeloggt ist, erhält er die Möglichkeit, zur Gerätezuordnung zu wechseln. Der Raum ‚nicht zugeordnet' wird dazu entfernt und seine Geräte an die rechte Seite der Website verschoben, welche sich mitbewegt. So ist es einfacher, die Geräte den Räumen (via Drag&Drop) zuzuordnen. Zusätzlich ist es möglich, Räume mithilfe eines ‚+'-Buttons hinzuzufügen, zu benennen und mit einem ‚-' wieder zu entfernen (vgl. **Abbildung 12: Gerätezuordnung und Raumsteuerung**).

Abbildung 12: Gerätezuordnung und Raumsteuerung

Während des Drag&Drop wird automatisch erkannt, ob ein Sensor oder ein Aktor verschoben wird. In Abhängigkeit davon wird eine Trennzeile (zur sichtbaren Trennung von Aktoren und Sensoren), ebenso wie eine visuelle Anzeige für das Ablegen des Gerätes für den Nutzer angezeigt (vgl. **Abbildung 13: Sensorerkennung bei Drag&Drop**).

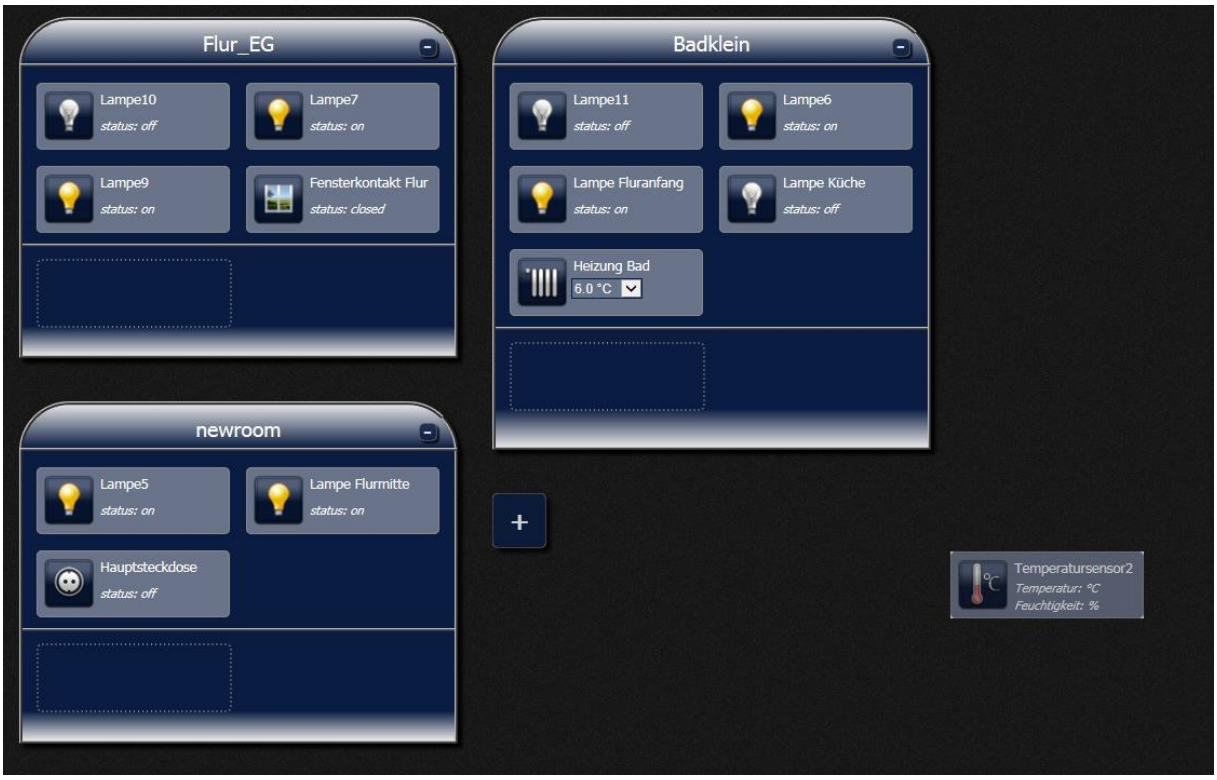

Abbildung 13: Sensorerkennung bei Drag&Drop

Der Administrator kann zusätzlich sämtliche Nutzer über die Benutzersteuerung verwalten. Dabei ist es möglich Nutzer anzulegen, zu löschen, ihr Passwort und E-Mail zu ändern, sowie eine Nutzerübersicht anzuzeigen (vgl. **Abbildung 14: Benutzersteuerung 'Registrierung'**).

Abbildung 14: Benutzersteuerung 'Registrierung'

Zusätzlich kann der Administrator über eine Kommandozeile direkt FHEM-Befehle abschicken. Nach der Eingabe der ersten drei Buchstaben wird der Nutzer dabei mit Autosuggest-Vorschlägen unterstützt (vgl. **Abbildung 15: Kommandozeile mit Autosuggest**

). Die Checkbox mit dem Haken rechts neben der Kommandozeile gibt an, ob die Befehle in der FHEM.cfg nach dem Abschicken gleich dauerhaft gespeichert werden sollen. Mit der Entertaste wird ein Befehl abgeschickt.

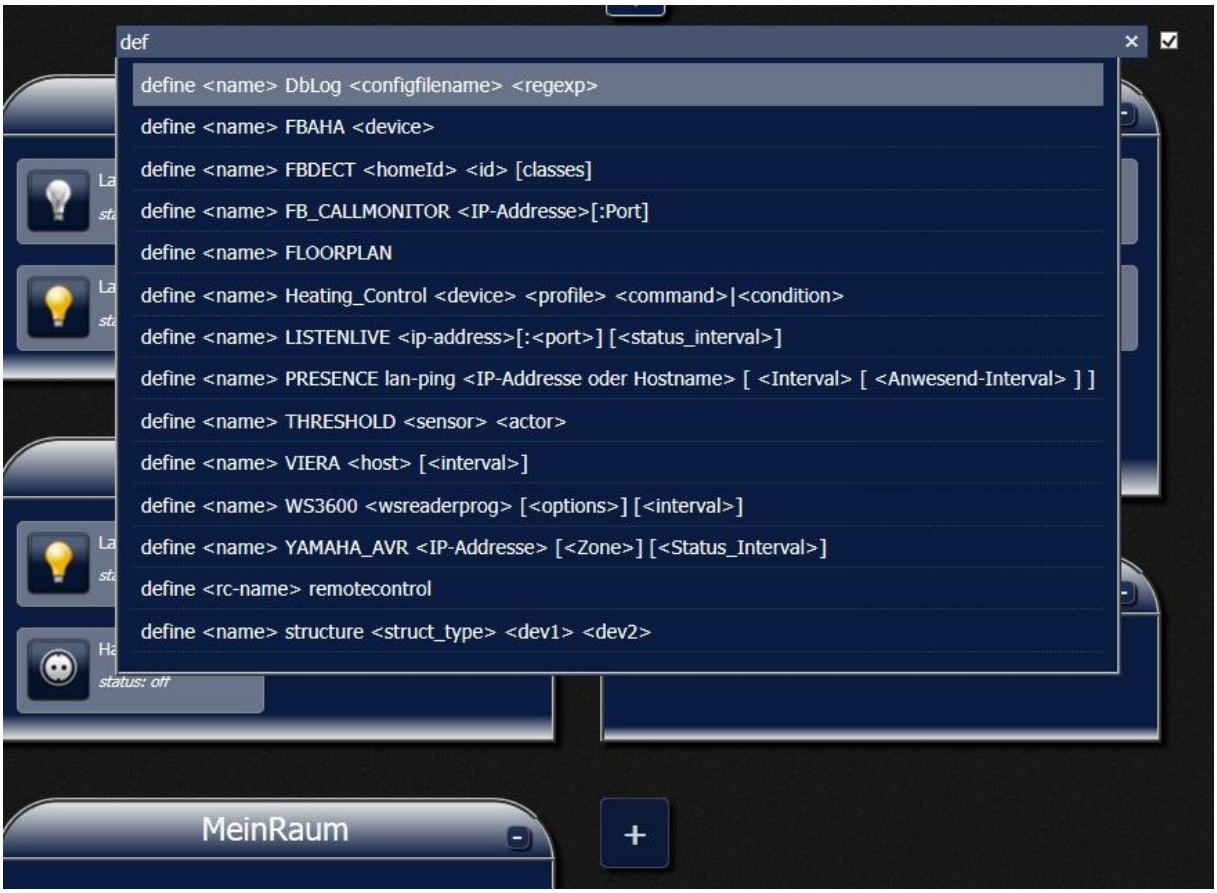

Abbildung 15: Kommandozeile mit Autosuggest

6.3.4 Verteilung der Last zwischen Server (PHP) und Client (JavaScript)

Durch das redundante Speichern der Objekte in PHP und JavaScript ist es möglich, die Last zwischen dem Server und Client zu verteilen. Momentan wird jedwede Userinteraktion clientseitig ausgeführt, welches in zukünftiger Implementierung für einen Performancegewinn sorgt. Erst die am Ende notwendigen Befehle zur Änderung des Models auf dem FHEM-Server werden an den Controller (PHP) gegeben, von diesem überprüft, angepasst und dann abgeschickt.

So ist es auch nicht nötig, ständig das gesamte Modell nachzuladen, sondern nur die Teile, welche sich wirklich verändert haben. Dazu empfiehlt es sich, in Zukunft noch ein automatisches (momentan ist es eventbasiert) Ajax-Polling zu implementieren.

Im Sinne der künftigen Erweiterbarkeit ist es also ohne Probleme möglich, die Floorplangenerierung, ebenso wie die Charting-Funktionalität in Abhängigkeit der Leistung des Servers oder Endgerät entweder client- oder serverseitig zu implementieren.

7. Ergebnisse und Bewertung

Im Folgenden werden die Ergebnisse des Forschungsseminares mit den Fragen der Zielstellung (*1.2 Zielstellung*) verglichen.

Welche besonderen funktionalen sowie nichtfunktionalen Anforderungen gelten im Bereich der Hausautomatisierung?

Im Rahmen des Forschungsseminares ist es gelungen, nichtfunktionale Anforderungen, eine Risikoanalyse und allgemeine, funktionale Anforderungen aus verschiedenen Quellen zu ermitteln. Anhand dieser wurden textuell und beispielhaft einige Anwendungsfälle formuliert. Dieser gesammelte Anforderungskatalog soll auch in Zukunft weiter bearbeitet und erweitert werden, um so auch bei sich in Zukunft ändernden Anforderungen aktuell zu bleiben.

Welche dieser Anforderungen lassen sich mit einem Web Frontend realisieren?

Anhand der Anwendungsfälle und den allgemeinen, funktionalen Anforderungen konnten spezielle Anforderungen für das Web Frontend abstrahiert werden. Dies wurde zusätzlich unter Berücksichtigung der technologischen Möglichkeiten betrachtet. Dieser gesammelte Anforderungskatalog soll auch in Zukunft weiter bearbeitet und erweitert werden, um so auch bei sich in Zukunft ändernden Anforderungen aktuell zu bleiben.

Welche Eigenschaften hat eine nutzerfreundliches Web Frontend unter Berücksichtigung verschiedener Nutzertypen im Bereich der Hausautomatisierung?

Im Rahmen des Forschungsseminars fiel auf, dass es im Bereich der Hausautomatisierung im Open Source Bereich oftmals nur Lösungen gibt, die für erfahrene Nutzer im Bereich der Programmierung oder Elektrotechnik nutzbar waren. Deshalb macht es Sinn, auch für unerfahrene Nutzer eine vereinfachte, grafische Bedienung zu ermöglichen.

Auf Basis der Analyse (siehe *3. Anforderungsanalyse*) und der technologischen Betrachtungen (siehe *4. Technologische Betrachtungen*) war es nun möglich, einen Entwurf für das Web Frontend zu erstellen (siehe *5. Entwurf*). Auf Basis des Entwurfes fand eine prototypische Implementierung (siehe 6. Prototypische Implementierung) als Grundlage für künftige Entwicklungen statt.

8. Schlussbemerkung und Ausblick

Zusammenfassend ist zu sagen, dass durch die Anforderungsanalyse und die technologischen Betrachtungen erst ein sinnvoller Entwurf eines Web Frontend für zukünftige Nutzer mit verschiedener Erfahrung im Bereich der Hausautomatisierung und Programmierung möglich war.

Der vorgestellte Entwurf (siehe *5. Entwurf*) ist daher eine auf den Anforderungen und technologischen Betrachtungen basierende Konzeption eines anwenderfreundlichen Web Frontend. Der Entwurf wurde so konzipiert, dass man auf dieser Basis sämtliche bereits gesammelten Anforderungen abdecken kann. Es wird zeitgleich auch für Nachfolger gut möglich sein, neue Anforderungen in den Entwurf aufzunehmen und zu implementieren. Zukünftig ist es sinnvoll, eine Betrachtung der Implementierung auf einer FRITZ!Box oder Raspberry Pi durchzuführen.

Die nächsten Ziele einer nachfolgenden Gruppe sollten das automatische Ajax-Polling, sowie die Implementierung der Regelverarbeitung sein, damit die Aufgaben mit höchster Priorität abgeschlossen sind. Darauffolgend wäre das Einbauen des Chartings (Diagrammanzeige bei Klick auf Sensoren) und Floorplans ein nächster, grundlegender Schritt. Im CSS des Web Frontends gibt es zusätzlichen Optimierungsbedarf.

Im Sinne der Floorplangenerierung und ~steuerung sollte zusätzlich noch eine genauere Betrachtung des YAF (Yet Another Floorplan) stattfinden (siehe *4.4 Möglichkeiten zur Floorplanerstellung*). Weiterhin kann über eine zukünftige Nutzung von Servlets und ggf. Webservices nachgedacht werden. Dazu wäre eine genauere Betrachtung der Java-Schnittstelle (siehe *4.6 Sonstige Betrachtungen*) nötig.

Datenbanktechnisch ist zusätzlich eine genauere Auseinandersetzung mit dem Thema ‚Pepper' möglich. Da bisher die gegebenen Sicherheitsalgorithmen keine Hinzugabe eines Peppers zum Salt ermöglicht haben und die Gruppe Web Frontend nicht den Algorithmus verändern möchte (Sicherheitsrisiko), wurde bisher davon abgesehen.

Zusätzliche Sicherheitsaspekte (verschlüsselte Telnet-, sowie Funkverbindungen) waren nicht Teil des Forschungsseminares, sollten aber in Zukunft bedacht und eingebaut werden.

V. Literaturverzeichnis

[BERN13] Henry Berndt - Guck mal wer da spricht - Sächsische Zeitung, 2. (30. 05 2013). Guck mal, wer da spricht! *Sächsische Zeitung*, 3.

[EFFE13] Frank Effenberger - Diskussion im FHEM-Forum, 2. (20. 05 2013). *FHEM-Forum*. Abgerufen am 03. 06 2013 von http://forum.fhem.de/index.php?t=msg&th=12910&start=0&rid=0

[FHEM13] Homepage für freundliche Hausautomatisierung und Energiemessung, 2013. (kein Datum). *FHEM*. Abgerufen am 03. 06 2013 von http://fhem.de/fhem.html

[FHWI13a] Grundriss mit Buttons aus der FHEMWiki, 2013. (kein Datum). *Fhemwiki*. Abgerufen am 04. 06 2013 von Grundriss mit Buttons: http://www.fhemwiki.de/wiki/Grundriss_mit_fhem-buttons

[FHWI13b] Neues Charting Frontend aus der FHEMWiki. (12. 05 2013). *Fhemwiki*. Abgerufen am 04. 06 2013 von Neues Charting Frontend: http://www.fhemwiki.de/wiki/Neues_Charting_Frontend

[FIFF13a] Frauenhofer Institut für Fabrikbetrieb und ~automatisierung, 2013. (kein Datum). *IFF Frauenhofer Institut*. Abgerufen am 03. 06 2013 von Gesundheit: http://www.iff.fraunhofer.de/de/anwendungsfelder/gesundheit.html

[FIFF13b] Frauenhofer Institut für Fabrikbetrieb und ~automatisierung, 2013. (kein Datum). *IFF Frauenhofer Institut*. Abgerufen am 03. 06 2013 von Produktion: http://www.iff.fraunhofer.de/de/anwendungsfelder/produktion.html

[FIML13a] Frauenhofer Institut für Materialfluss und Logistik zum Internet der Dinge, 2013. (kein Datum). *Internet der Dinge*. Abgerufen am 03. 06 2013 von Frauenhofer Institut: http://www.internet-der-dinge.de/de/wasistdasinternetderdinge.html

[FIML13b] Frauenhofer Institut für Materialfluss und Logistik, 2013. (kein Datum). *Das Internet der Dinge*. Abgerufen am 03. 06 2013 von Frauenhofer Institut: http://www.internet-der-dinge.de/

[GEKI13] FHEM JAVA-Schnittstelle, G. 2. (06. Februar 2013). *FHEM Java-Schnittstelle*. Abgerufen am 27. Dezember 2013 von FHEM Java-Schnittstelle: http://forum.fhem.de/index.php/topic,10867.0.html

[HAPK05 S.16] Hapke, R. (28. 04 2005). *Grundsätze der Dialoggestaltung nach DIN EN 9241-10*. Abgerufen am 03. 06 2013 von Hochschule für Technik, Wirtschaft und Kultur Leipzig: http://www.imn.htwk-leipzig.de/~waldmann/edu/ss05/se/talk/rhapke/hapke.pdf

[HAWI13] HomeMini Beschreibung auf Hansemanns Wiki, 2013. (11. 01 2012). *Hansemanns Wiki*. Abgerufen am 04. 06 2013 von HomeMini: http://home.nendzig.net/wiki/index.php/HomeMini

[HUUV10] Halbjahresbericht 2010 zur UMTS-Abdeckung, 2. (07. 07 2010). *HDSPA / UMTS Verfügbarkeit*. Abgerufen am 03. 06 2013 von http://www.hsdpa-umts-verfuegbarkeit.de/blog/2010/07/07/halbjahresbericht-2010-zur-umts-verfuegbarkeit-70-netzabdeckung-erreicht/

[HUUV13] Meldung der Bundesnetzagentur über 325 Millionen umts Nutzer im Jahr, 2. (23. 05 2013). *HSDPA / UMTS Verfügbarkeit*. Abgerufen am =§. 06 2013 von http://www.hsdpa-umts-verfuegbarkeit.de/blog/2013/05/23/bundesnetzagentur-meldet-325-millionen-umts-nutzer-im-jahr-2012/

[KALIE13] Andy Kalies Beispiele für Hausautomatisierung, 2. (kein Datum). *Hausbau Blog*. Abgerufen am 03. 06 2013 von http://www.haus-bau-blog.de/haus-technik/moeglichkeiten-hausautomatisierung-beispiele-hausautomation-haussteuerung/

[KÖNI13] König - Website von FHEM, 2. (2013). *FHEM*. Abgerufen am 04. 06 2013 von Freundliche Hausautomatisierung und Energiemessung: http://fhem.de/fhem_DE.html

[LHÈG05] Philippe Le Hégaret, 2. (19. 01 2005). *World Wide Web Consortium*. Abgerufen am 03. 06 2013 von http://www.w3.org/DOM/

[PAPE13] JQuery-PHP-Ajax-Autosuggest. (8. August 2009). *Papermashup JQuery-PHP-Ajax-Autosuggest.* Abgerufen am 20. August 2013 von http://papermashup.com/jquery-php-ajax-autosuggest/

[PHPL13] PHPLogin.net. (26. Dezember 2013). *PHPLogin.* Abgerufen am 29. Dezember 2013 von http://www.php-login.net/

[RWE13], RWE Smarthome Website. (2013). *RWE Smarthome*. Abgerufen am 03. 06 2013 von https://www.rwe-smarthome.de/web/cms/de/468182/smarthome/informieren/was-ist-rwe-smarthome/komfort-steigern/

[SRCF13] fhzctrl von Sourceforge, 2013. (kein Datum). *Sourceforge*. Abgerufen am 04. 06 2013 von fhzctrl: http://fhzctrl.sourceforge.net/

[W3CS13] W3CSchools. (20. August 2013). *W3CSchools.* Abgerufen am 30. August 2013 von www.w3schools.com/

[WEMA13] Daniel Weisensee Markus Mangei YAF-Projekt, 2. (9. November 2013). *Yet Another Floorplan Wiki*. Abgerufen am 12. Dezember 2013 von http://www.fhemwiki.de/wiki/YAF

[WIKI13] Aktor, h. (26. 03 2013). *Wikipedia*. Abgerufen am 03. 06 2013 von http://de.wikipedia.org/wiki/Aktor

[YAF13] Screenshot FhemWiki zu YAF, 2. (11. September 2013). *Fhemwiki.* Abgerufen am 27. Dezember 2013 von http://www.fhemwiki.de/w/images/e/e1/YAF_Beispielansicht.png

VI. Anlagen

Anlage 1: Nichtfunktionale Anforderungen

Anforderung	Kurzbeschreibung	Lösung
Sicherheit	Maßnahmen zur Vermeidung von gefährlichen Zuständen.	Authentifizierung, Verschlüsselung.
Funktionserfüllung	Übereinstimmung funktionale Anforderung und realisierte Funktionalität.	Testen der Anwendung.
Benutzbarkeit(Usability)	Alle Eigenschaften, die dem Benutzer ein angenehmes, einfaches und effizientes Arbeiten ermöglichen.	Klare Strukturierung und Komposition der Website.
Effizienz	Maß für die Inanspruchnahme von Betriebsmitteln bei gegebenem Funktionsumfang.	Kombination von PHP und JavaScript.
Effektivität	Maß für die Wirksamkeit, Verhältnis von erreichtem zu definiertem Ziel.	Regelmäßiges Überprüfen der realisierten Funktionalität in Hinblick auf Anforderungsanalyse.
Zuverlässigkeit	Wahrscheinlichkeit, mit der das System unter festgelegten Qualitätsmerkmalen und Rahmenbedingungen die funktionalen Anforderungen erfüllt.	Festlegen der Qualitätsmerkmale und testen dieser.
Erweiterbarkeit	Maß für die Schwierigkeit des Einfügens einer neuen Funktionalität.	Konzipierung eines modularisierten Programms.
Änderbarkeit	Maß für die Schwierigkeit eine vorhandene Funktionalität anhand von neuen Qualitätsmerkmalen zu ändern.	Konzipierung eines modularisierten Programms.
Portabilität	Plattformunabhängigkeit/Übertragbarkeit.	Nutzung einer plattformunabhängigen Sprache.
Kompatibilität	Kommunikation mit FHEM-Server.	Nutzung von Telnet Port 7072.
Individualisierbarkeit	Möglichkeit des Nutzers, das Erleben der Seite zu individualisieren.	Eigene Themes, Erstellen eigener Geräte.
Selbstbeschreibungsfähigkeit	Maß für die Offensichtlichkeit der Steuerung der Benutzeroberfläche, selbsterklärende Funktionalitäten.	Testen durch Nutzer, Einschätzung der Offensichtlichkeit der Steuerung.
Lernförderbarkeit	Maß für die Lernkurve, mit der der Nutzer den Umgang mit dem System	Einbauen einer Hilfe.

	lernt.	
Erwartungskonformität	Maß für die Übereinstimmung der Erwartung eines Nutzers mit implementierter Funktionalität.	Diskussion und Test durch Endnutzer
Wartbarkeit	Maß für die Möglichkeit von Fehlerreparatur, Erweiterung und Änderungen am System	In die Hände von FHEM-Community geben, gute Doku, Forschungsseminar nach uns
Usability	Maß für die Bedienbarkeit des Web Frontend für Menschen mit Behinderungen (Sehschwäche, etc.)	Test mit Menschen mit Behinderungen

Anlage 2: Allgemeine, funktionale Anforderungen

Nr	Name der Anforderung	Beschreibung	Merkmale	Sonstiges (Einordnung)
1	Steuerung der Rolläden/ Jalousien via Web Frontend.	Nutzer kann via Endgerät die Rollläden/Jalousien hoch- und runterfahren lassen.	Stufenweises herunter/hochfahren sollte möglich sein.	Steuerung
2	Steuerung der Rolläden/ Jalousien über Aktoren/ Sensoren.	Rollläden/Jalousien reagieren bei bestimmten Sensorwerten.	Gruppensteuerung mehrerer Rollläden/Jalousien, Zeitsteuerung, Sonneneinstrahlung messen.	Steuerung / Regelverarbeitung
3	Lichtsteuerung via Web Frontend.	Nutzer kann via Endgeräte die Lichter steuern.	Schalten, dimmen	Steuerung
4	Lichtsteuerung via Aktoren/ Sensoren.	Licht kann bei bestimmten Sensorwerten gesteuert werden.	Zeitsteuerung, Bewegungsmelder, dimmen, schalten	Steuerung / Regelverarbeitung
5	Heizungs-steuerung via Web Frontend.	Nutzer kann via Endgeräte die Heizung steuern.	Schalten, Temperieren	Steuerung (auch Fußbodenheizung)
6	Heizungs-steuerung via Aktoren/ Sensoren.	Heizung kann bei bestimmten Sensorwerten gesteuert werden	Schalten, temperatur- und zeitabhängiges Temperieren	Steuerung / Regelverarbeitung (auch Fußbodenheizung), wenn gelüftet wird Heizung ausschalten.
7	Steuerung der Steckdosen via Web Frontend.	Nutzer kann via Endgerät Steckdosen schalten.	An- und Ausschalten von Steckdosen oder Steckdosen-gruppen.	Steuerung
8	Steuerung der Steckdosen via Sensoren/ Aktoren	Steckdosen werden bei bestimmten Sensorwerten aktiviert/deaktiviert.	Zeitsteuerung, Deaktivierung bei Gefahren (Feuer, Rauch, Wasser).	Steuerung / Regelverarbeitung
9	Garagentor-steuerung über Aktoren/ Sensoren.	Garagentor kann bei bestimmten Sensorwerten geöffnet/ geschlossen werden.	RFID-Signal gekoppelt mit Bewegungsmelder, damit man nicht manuell es öffnen oder schließen muss.	Steuerung/Regelver-arbeitung
10	Abruf und Statusänderung	Bei bestimmten Sensorwerten/	Abruf aktueller Energieverbrauch,	Information

	per SMS/eMail	Zuständen kann der Nutzer informiert werden.	Temperatur, etc.	
11	Steuerung via PC, Tablet und Smartphone.	Die Anwendung kann über diverse Endgeräte genutzt werden.		Steuerung, grundlegend
12	Wetteranzeige	In der Steuerung ist eine Wetteranzeige integriert.		Information
13	Notizblock-funktion	Man kann Notizen anlegen.	Notizen über Hausplanung / baldige Änderungen	Information
14	Nutzung der Alarmanlage.	Alarmanlage reagiert auf Sensoren.	Unbefugtes Betreten der Wohnung wird gemeldet, Zeitsteuerung.	Sicherheit, Steuerung
15	Nutzung Brand/Rauch- und Gasmelder.	Meldesystem reagiert auf Sensoren.	Erkennung lebensgefährlicher Zustände und geben von Alarm	Sicherheit, Steuerung
16	Nutzung Überwachungs-kamera.	Livestream / Bilder pro Sekunde von Überwachungs-kamera, Erkennungssystem.	Überwachung des Anwesens, Erkennung von neuen Personen	Sicherheit
17	Steuerung der Lüftung via Web Frontend.	Nutzer kann via Endgeräte die Belüftungssysteme Steuerung.	An/Aus, bestimmte Zwischenwerte.	Steuerung
18	Steuerung der Lüftung via Aktoren/ Sensoren.	Lüftung kann auf Sensorwerte reagieren.	Zeitsteuerung, Temperatursteuer-ung, Luftfeuchte-steuerung.	Steuerung, Regelverarbeitung
19	Verschlüsselte Authentifizierung	Nutzer kann sich nur mit Username/Passwort anmelden.	Erschwernis für Außenstehende, in das System zu kommen.	Sicherheit, Webseitig
20	Steuerung der Wasserzufuhr via Aktoren/Sensoren	Wasserzufuhr kann bei gefährlichen Zuständen unterbrochen werden.	Bei Wasserrohrbruch wird die Wasserzufuhr unterbrochen.	Sicherheit, Regelverarbeitung
21	Sensorsteuerung an Türen und Fenstern zur Öffnung/Schließu	Nutzer kann mit einem Endgerät Türen und Fenster im Haus	Eingangstür, Zimmertüren, Kühlschranktür, Geschirrspüler,	Steuerung

	ng via Web Frontend.	öffnen/schließen und sich den Status anzeigen lassen.	Fenster auf/zu.	
22	Sensorsteuerung an Türen und Fenstern via Sensoren.	Türen und Fenster reagieren auf Sensorwerte.	Schließung Kühlschranktür nach x Minuten, wenn gelüftet wird (Fenster / Türen offen) sollen Heizungen aus gehen.	Steuerung, Regelverarbeitung
23	Sensorsteuerung an Töpfen/Gefäßen auf dem Herd	Sensoren an Töpfen/Herd reagieren bei Überkochen	Herd abstellen, wenn Inhalte im Kochtopf überlaufen.	Steuerung, Regelverarbeitung
24	Steuerung Bewässerungs-system via Web Frontend.	Nutzer kann via Endgeräte seine Pflanzen bewässern.	Bewässerung an/aus, je nach Pflanze.	Steuerung
25	Sensorsteuerung an Pflanzen auf Balkon	Bewässerungs-system reagiert auf Sensorwerte.	Bei bestimmter Erdfeuchte/ Sonnenein-strahlung / Außentemperatur oder zeitlich wird bewässert.	Steuerung, Regelverarbeitung
26	Luftfeuchtesteuer ung via Web Frontend.	Nutzer kann via Endgeräte seine Luftfeuchte im Haus einstellen.	Manuelle Erhöhung/Sen-kung der Luftfeuchte.	Steuerung
27	Luftfeuchte-steuerung via Sensoren	Luftfeuchte-steuerung reagiert auf Sensoren.	Automatische Erhöhung/ Senkung der Luftfeuchte.	Steuerung, Regelverarbeitung
28	Aufenthalts-erkennung.	Sensoren merken, ob Personen anwesend sind.	Alarm, wenn alte Menschen sich x Minuten nicht bewegen.	Steuerung, Regelverarbeitung
29	Steuerung Fernseher via Aukustik.	Wenn geklatscht wird oder ‚Befehl' gesagt wird, geht Fernseher an.	Reaktion auf Sprachmuster.	Steuerung, Regelverarbeitung
30	Anmeldung als normaler Nutzer.	Nutzung für ‚Alltägliches'.	Normaler Nutzer hat begrenzte Rechte.	Sicherheit
31	Anmeldung als Administrator.	Nutzung zum Anlegen von Geräten und Zuordnung zu	Administrator hat erweiterte Rechte.	Sicherheit

		Räumen, Befehle absetzen.		
32	Anlegen von Räumen.	Anlegen von Räumen.	Namen geben	Steuerung/Regulierung
33	Zuordnung von Geräten zu Räumen.	Gerät wird einem Raum zugeordnet.	Gruppierung	Steuerung/Regulierung
34	Löschen von Räumen.	Raum wird gelöscht.	Zuordnungen werden ebenso gelöscht.	Steuerung/Regulierung
35	Anlegen von Geräten.	Ein Gerät wird angelegt.	Gerät benennen.	Steuerung/Regulierung
36	Änderung der Werte von Geräten.	Änderung der Werte von Geräten.	z.B. Änderung Name, Werte.	Steuerung/Regulierung
37	Anzeige der Eigenschaften eines Gerätes	Anzeige von spezifischen Daten.		Steuerung/Regulierung
38	Löschung eines Gerätes.	Löschen eines Gerätes.	Löschung aus Raumzuordnung.	Steuerung/Regulierung
39	Erstellung eines Floorplans.	Anlegen eines Floorplans.	Namen geben.	Steuerung/Regulierung
40	Modifizierung eines Floorplans.	Änderung von Werten, Steuerung Aktoren.		Steuerung/Regulierung
41	Löschen eines Floorplans.	Löschen des Floorplans.	Aufhebung eventueller Zuordnungen	Steuerung/Regulierung
42	Analyse von Sensorwerten.	Analyse von Sensorwerten.		Analyse
43	Umschalten zwischen verschiedenen Modis.	Anfänger / Expertenmodus, Urlaubs- und Alltagsmodus	Verstecken von Begriffen wie FS20 oder Cul_HM im Anfängermodus.	Steuerung/Regulierung
44	Sensorsteuerung bei Wassersensoren.	Alarm, wenn Wasser auf Boden in Bad oder Küche ist (Überlaufschutz).	Gegebenfalls Wasserzufuhr sperren, Warnung ausgeben (Hausstimme).	Sicherheit
45	Havarieknopf bei Problemen	Verständigung einer Notfallzentrale durch Drücken eines Notfallknopfes.	Alternativ sehr langes Drücken eines Lichtschalters.	Sicherheit
46	Pairing-Verfahren starten	Implementierung eines Autocreate-Buttons, mit dem via FHEM die	Ggf. auch ‚Wizard' zur Unterstützung des Autocreates	Steuerung

		Gerätekommunikation abgewickelt wird.	anbieten.	

Anlage 3: Abstrahierte, funktionale Anforderungen

Grundlegendes:

(1) Erstellung einer Komposition, die den Anforderungen genügt.

(1)Finden einer Farbgebung, die den nichtfunktionalen Anforderungen genügt.

Steuerung (durch Login getrennter „Adminmodus“) :

(1)Manuelles Anlegen eines Gerätes

(1)Absetzen von Befehlen über eine Kommandozeile

(1)Login über Adminpassword

(1)Anzeige der Eigenschaften eines Aktors

(1)Gruppieren von Aktoren und Sensoren in Räumen

(1) Anlegen/Umbenennen/Löschen von Räumen

(2)Gruppieren von Aktoren zu Steuerungsgruppen bei gleicher Steuerungsmöglichkeit

(1)Auswahl der Steuerungsmöglichkeiten für einen Aktor (Default durch Typ) → Dimmen, Schalten, (Prozentangabe), Temperaturangabe

(1)Anzeige der Eigenschaften von Sensoren

(4)Verknüpfen des Aktorverhaltens mit einem Sensor (möglichweise n zu m)

(3)Benachrichtigung des Nutzers bei bestimmten Sensorverhalten oder Aktorzustand.

(1)Konfiguration von Regeln für einen Aktor oder eine ganze Aktorengruppe (z.B. zeitgesteuert)

(4)Aktivierung des Pairings

(4)Einfügen und Positionieren von Benutzer eigenem HTML Code

(2)Charting Konfigurieren

(4)Floorplan Konfigurieren (Konfigurationsmodus)

Charting:

(2)Erstellen eines Charts

(2)Zuweisen von Sonsoren zu einem Chart

(3)Einstellen der Aggregationsstufe für die Daten. (Ausführung im Hintergrund)

(2)Zoomen in einem Chart

(2)Verschieben des Charts auf der Zeitachse

(2)Erweiterte Interaktion mit dem Chart

(2)Konfiguration der Skalenbereiche

Interaktionsmodus:

(1) Anzeige aller Aktoren

(1) Änderung der Aktorenzustände

(2) Anzeige aller Charts

(3) Automatische Aktualisierung der Charts

(1) Anzeige aller Räume

(3) Automatische Aktualisierung der Aktorenzustände

(4) Anzeige einer Floorplanansicht (Interaktionsmodus)

Floorplanansicht (Interaktionsmodus):

(4)Anzeige des Floorplans

(4)Änderung der Aktorenzustände

(4)automatische Aktualisierung der Zustände eines Aktors

Floorplanansicht (Konfigurationsmodus):

(4)Einfügen eines Floorplans

(4)Einfügen und Positionieren der Aktoren oder Aktorengruppen über die Web Oberfläche

(4)Anpassen der Aktoren in ihrer Darstellung (Verschieden Typen)

(4)Einfügen und Positionieren von Benutzer eignem HTML Code

Es ist ebenso eine grafische Darstellung der Interaktion des Administrators mit dem Endsystem vorhanden (vgl. **Abbildung 16: Interaktion des Administrators mit dem System**).

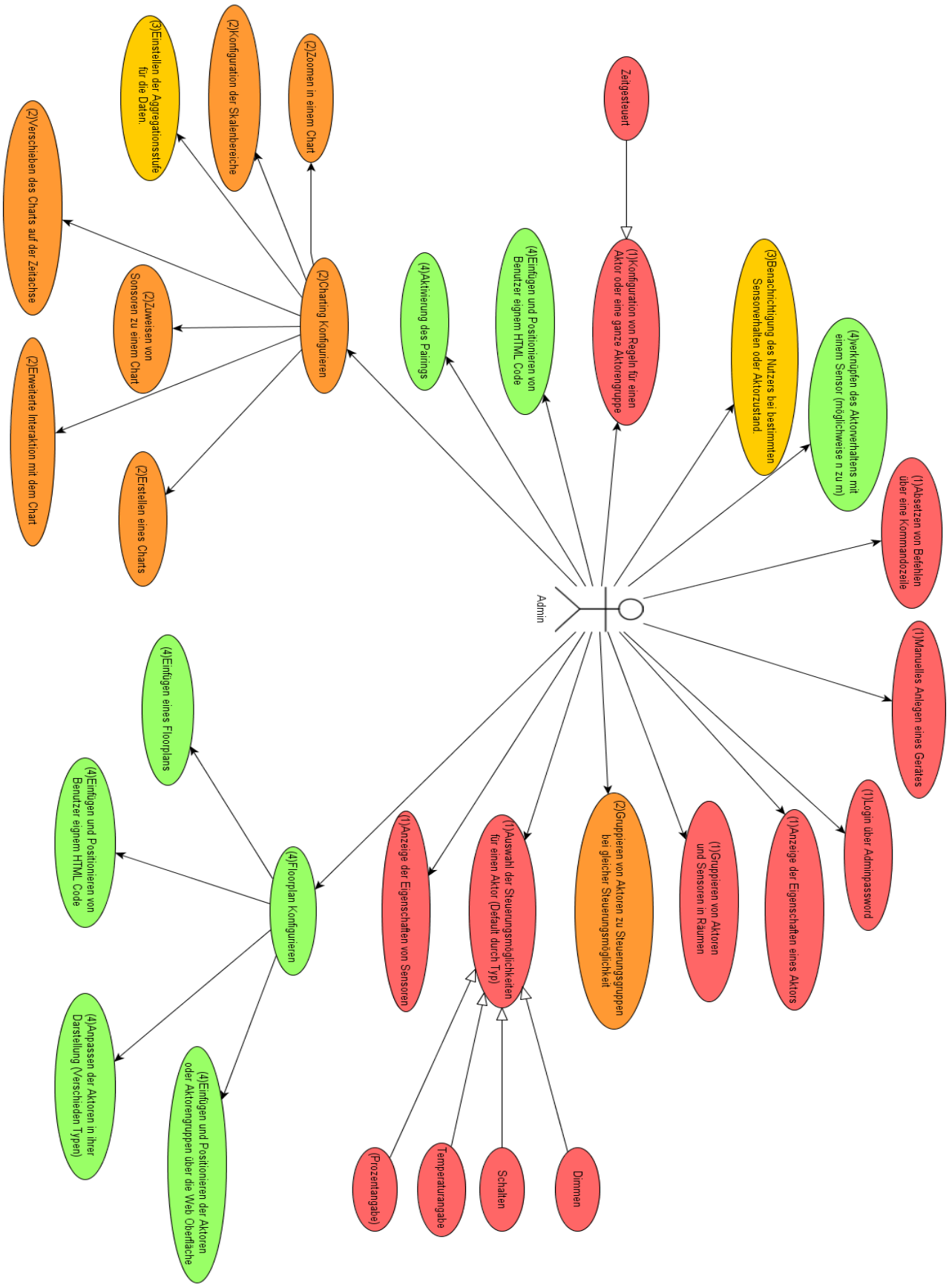

Abbildung 16: Interaktion des Administrators mit dem System

Anlage 4: JavaScript Charting Bibliotheken

Die JavaScript-Bibliotheken wurden anhand der Kriterien der funktionalen und nichtfunktionalen Anforderungen (und unter Berücksichtigung von Einhaltung von Lizenzen, damit alles Open Source bleibt) bewertet. Dabei wurde vor allem ein Wert darauf gelegt, wie das Erlebnis eines Nutzers ist. Grün hinterlegte Bibliotheken wurden von als positiv und für diese Anforderungen verwendbar eingestuft.

Open Source (ohne Zusatzbibliothek):

Bluff: http://bluff.jcoglan.com/

- Leichte Interaktion möglich
- Optik nicht überzeugend
- Funktionsumfang gering

Canvas 3D Graph: http://dragan.yourtree.org/code/canvas-3d-graph/

- Nur 3D

CanvasXpress: http://canvasxpress.org/line.html

- Leichte Interaktion möglich
- Optik nicht überzeugend

ccchart: http://ccchart.com/#2

- Anwendung von Websockets
- Keine Interaktion möglich
- Optisch recht gut

D3: http://projects.flowingdata.com/life-expectancy/

- Ausgiebige Interaktion möglich
- Optisch recht gut

- Möglicherweise aufwendige Implementierung

dhtmlxChart (also commercial license) : http://www.dhtmlx.com/docs/products/dhtmlxChart/index.shtml

- Keine Interaktion in den Beispielen
- Optisch überzeugend

dygraphs : http://dygraphs.com/

- Sehr ausgiebige Interaktion möglich
- Optisch überzeugend (professioneller Eindruck)
- Einfach zu implementieren

Flotr2: http://www.humblesoftware.com/flotr2/

- Recht gute Interaktion möglich
- Optisch überzeugend

gRaphaël: http://g.raphaeljs.com/

- Leichte Interaktion möglich
- Optisch nicht überzeugend

Ico: http://alexyoung.github.io/ico/

- Keine Interaktion möglich
- Optisch nicht überzeugend

JSXGraph: http://jsxgraph.uni-bayreuth.de/wp/examples/

- Leichte Interaktion möglich
- Optisch nicht überzeugend

Google Charts: https://developers.google.com/chart/interactive/docs/gallery?hl=de

- Keine Interaktion möglich
- Nur Vektorgrafiken
- Optisch in Ordnung

Protovis: http://mbostock.github.io/protovis/ex/zoom.html

- Gute Interaktionsmöglichkeiten
- Optisch überzeugend
- Aufwändige Implementierung

Open Source (mit Zusatzbibliothek):

Elycharts (jQuery) : http://elycharts.com/examples

- Gute Interaktion
- Optisch nicht überzeugend

Flot (jQuery) : http://www.flotcharts.org/flot/examples/tracking/index.html

- Ausgiebige Interaktion möglich
- Optisch überzeugend

jqPlot (jQuery) : http://www.jqplot.com/tests/data-renderers.php

- Keine Interaktion möglich
- Optisch überzeugend

MilkChart (MooTools): http://www.brettdixon.com/demos/charts.html

- Keine Interaktion möglich
- Optisch überzeugend

PlotKit (MochiKit): http://www.liquidx.net/plotkit/

- Keine Interaktion möglich
- Optisch nicht überzeugend

Nicht Open Source:

FusionCharts : http://www.fusioncharts.com/demos/business/

- Auf Dashboards spezialisiert
- Kein Gewinn für das Projekt gegenüber Open Source

Highcharts : http://www.highcharts.com/demo/

- Ausgiebige Interaktion möglich
- Optisch sehr ansprechend

JS Charts : http://www.jscharts.com/examples

- Ausgiebige Interaktion möglich
- Optisch recht gut

Anlage 5: Vorhandene Software-Lösungen des Marktes

IP-Symcon:

- Ein Test des Web Frontend ist möglich über: http://www.webfront.info/
- Die Erstellung fand mithilfe der Google API's statt.
- Durch starke Nutzung von JavaScript ist eine hohe Interaktivität mit dem System möglich.
- Die Last liegt auf Clientseite, da die Seitengenerierung im JavaScript stattfindet.
- Die Seite zeichnet sich durch eine starke Nutzung von Ajax aus.
- Miteinander abgestimmte Farbverläufe sorgen für ein optisch gutes Nutzererlebnis.
- Die Menüs sind hierarchisch gegliedert.
- Die Bedienung ist selbsterklärend und intuitiv.
- Die Interaktivität mit dem System ist an mehreren Stellen geben, insgesamt jedoch eher gering.
- Es findet eine klare Trennung der Bedienung vom der Konfiguration statt.

RWE Smarthome:

- Ein Beispielvideo zur Nutzung von RWE Smarthome ist unter folgendem Link zu finden: http://www.youtube.com/watch?v=84RRXyy-AkE
- Diese Lösung wurde in Silverlight geschrieben.
- Die Steuerung basiert auf Drag & Drop Mechaniken und ist sehr intuitiv.
- Die Navigation basiert auf einem komplett eigenen Konzept, was jedoch leicht verständlich ist.
- Diese Lösung ist sehr interaktiv und ansprechend.

Homematic:

- Ein Beispielvideo zur Nutzung von Homematic ist unter folgendem Link zu finden: http://www.youtube.com/watch?v=WR0ftonatVc
- Die Seite wirkt sehr statisch und ist optisch nicht ansprechend.
- Durch eine funktionale Ausrichtung ist das Frontend recht intuitiv zu bedienen.
- Eine freie Regelkonfiguration ist ohne zusätzliches Coding möglich.

Homeputer:

- Die Software ist unter folgendem Link zu finden (180€): http://www.contronics.de/shop/Zentralen-und-Software/Homeputer-CL-Studio-Software-fuer-HomeMatic
- Diese Lösung bietet die Generierung von Webseiten für einen eigenen Server an.
- Diese Webseiten sind sehr primitiv und funktional gehalten, sodass diese optisch gesehen nicht ansprechend sind.
- Es gibt eine iPhone-App (Pocket Control CL) für das Programm.
- Dieser Anwendung ist ein Desktop-Programm und kein Web Frontend.

EZcontrol:

- EZcontrol beinhaltet einen integrierten Web Server.
- Die Optik des Web Frontend sticht nicht sonderlich hervor.
- Die Navigation über eine Art von Karteireiter.
- Es gibt ein sehr großes App Angebot für EZcontrol, unter anderem:
    - "monitor4home" für iPhone und iPad.
    - "XS1 Heimautomatisierung" für Android-Systeme.
    - „XS1 Home Control“ für Windows Phone

Marmitek:

- Die Software „X10 ActiveHomePro“ ist unter folgendem Link zu finden (ca. 50 €): http://www.activehomepro.com/activehome-pro.html
- Es sind Apps für IPhone und Android erhältlich, diese heißen "X10 - Commander".
- Es gibt keinen Web Service zur Steuerung und ist damit eine Desktopanwendung.