Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Masterarbeit

Sven Boris Bornemann

Entwicklung eines kontextsensitiven Berechtigungssystems für Smart Homes

Fakultät Technik und Informatik
Studiendepartment Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Sven Boris Bornemann

Entwicklung eines kontextsensitiven Berechtigungssystems für Smart Homes

Masterarbeit eingereicht im Rahmen der Masterprüfung

im Studiengang Master of Computer Science
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. rer. nat. Kai von Luck
Zweitgutachter : Prof. Dr. rer. nat. Gunter Klemke

Eingereicht am: 02.12.2013

Sven Boris Bornemann

Thema der Arbeit
Entwicklung eines kontextsensitiven Berechtigungssystems für Smart Homes

Stichworte
Smart Home, Smart Environments, Complex Event Processing, CEP, Role Based Access Control Model, RBAC, Berechtigungssysteme, Context awareness

Kurzzusammenfassung
Der Mensch bewegt sich heutzutage in Umgebungen, in denen er von technischen Gegenständen umgeben ist. Die zunehmende Kommunikationsfähigkeit dieser Geräte ermöglicht die Entwicklung sogenannter Smart Environments. Diese Entwicklung ist vor allem in den Wohnumgebungen des Menschen zu spüren. Ziel dieser Arbeit war die Entwicklung eines Berechtigungssystems für diesen sensiblen und komplexen Lebensbereich. Um die alltäglichen Situationen abbilden zu können, wurden das Berechtigungssystem an eine Kontexterkennung gekoppelt. Dies ermöglicht zum einen die Definition statischer Regeln, zum anderen wird die Zuweisung von Berechtigungen in Abhängigkeit des vorherrschenden Kontextes ermöglicht.

Sven Boris Bornemann

Title of the paper
Development of a context-sensitive access control system for smart homes

Keywords
Smart Home, Smart Environments, Complex Event Processing, CEP, Role Based Access Control Model, RBAC, Access Control Systems, Context awareness

Abstract
Nowadays Humans live in environments surrounded by technological artifacts. The increasing communication capabilities of these artifacts allow the uprising of the so called "Smart Environments". This development is specifically felt in domestic living environments. The aim of this Thesis is the design of an authorization system for these delicate and complex areas of life. In order to better reflect the real life situations an access control system is annotated with measures of context awareness. This enables the definition of static rules and the assignment of permissions depending on the prevailing context.

Danksagung

Ich möchte mich an dieser Stelle ganz herzlich bei allen Menschen bedanken, die mich bei der Erstellung dieser Arbeit unterstützt haben.
Besonders möchte ich mich aber bei Prof. Dr. rer. nat. Kai von Luck und Prof. Dr. rer. nat. Gunter Klemke für Ihre Unterstützung bei der Realisierung meiner Projekte während des gesamten Studiums bedanken.

Inhaltsverzeichnis

1. Einleitung

Die Welt der Informatik unterliegt einem ständigen Wandel. Durch fortwährende Entdeckungen neuer Technologien und der Weiterentwicklung vorhandener Technologien und Standards dringen Computer in immer mehr Bereiche des menschlichen Lebens ein. Die Geräte werden immer kleiner und leistungsfähiger. Letzteres lässt sich vor allem an der Vielzahl von Funktionalitäten eines Gerätes erkennen. Des Weiteren besitzt jedes Gerät eine Netzwerkschnittstelle, welches die Verknüpfung von Funktionalitäten unterschiedlicher Geräte ermöglicht. Dadurch können sogenannte Smart Environments entstehen:

> *Smart environment as a small world where different kinds of smart device are continuously working to make inhabitants' lives more comfortable.*
>
> *Diane J. Cook und Sajal K. Das*
>
> *Cook und Das (2004)*

Smart Environments können verschiedene Ausmaße annehmen. Zum Beispiel die einer ganzen Stadt, in der Bewohner Zugriff auf unzählige Services haben. Diese reichen von Mobilitätskonzepten, über Unterhaltungsangebote, bis hin zu sozialen Leistungen. Für die Städte der Zukunft ist die Mobilität und das damit wachsende Verkehrsaufkommen eine große Herausforderung. Abhilfe können hier die Informationen verschiedene Services bieten, die einer Person den schnellsten und günstigsten Weg von A nach B aufzeigt. Informationen aus öffentlichen Verkehrsmitteln, Carsharing-Anbietern, Mitfahrzentralen, Autovermietungen, aktueller Verkehrslage und der momentanen Ausgangsposition der Person können helfen den optimalen Weg zu finden.
Smart Environments gibt es aber auch in einem viel kleinerem Maßstab, beispielsweise im Wohnumfeld des Menschen. In den Wohnungen und Häuser der Menschen nehmen die digitalen Gegenstände ebenfalls rapide zu. Mittlerweile sind so viele Gegenstände in einer Wohnumgebung mit Rechnerleistung versehen, dass der Mensch diese schon in sein tägliches Leben integriert hat und diese gar nicht mehr wahrnimmt. Diese Entwicklung prophezeite bereits Mark Weiser 1991 in seinem visionären Artikel *The Computer for the 21st Century*, vgl. Weiser (1991)
Einen großen Anteil an der Verbreitung tragen ebenfalls Smartphone- und Tablet-Geräte.

Durch diverse Schnittstellen und ihrer Anpassungsmöglichkeiten, bedingt durch das Touchdisplay, dienen sie als universale Steuerzentrale. Informationen können gebündelt dem Anwender präsentiert werden. Des Weiteren können mehrere Geräte bedient und gesteuert werden. Das Potential dieses universellen Steuergerätes zeigen diverse kommerzielle Produkte, wie beispielsweise: GIRA, ROCKETHOME, QIVICON oder Philips hue

1.1. Motivation

Die bereist heute in Wohnumgebungen vorhandenen elektronischen Geräte stellen dem Menschen unzählige Informationen und Funktionen zur Verfügung, unter anderem auch solche, die nicht für jedermann zugänglich sein sollen. Dies wirft die Fragen auf: *Wer oder was darf Zugriff auf welche Informationen und Funktionen erhalten?*
Das wer bezieht sich dabei auf die unterschiedlichen Parteien einer Wohnung, wie beispielsweise in Wohngemeinschaften oder Familien. Um private Daten zu schützen und gesellschaftliche Hierarchien abbilden zu können, wird eine Zugriffsteuerung benötigt. Die Einsatzgebiete sind sehr vielfältig, so könnte zum Beispiel in einer Familie Kinder geschützt werden, in dem dem Kinder keine schädliche Inhalte im Fernsehen präsentiert bekommen oder sie in Abwesenheit der Eltern die Haustür nur öffnen können, wenn der Besucher bekannt und auch gewünscht ist. Neben dem Aspekt des Personenschutzes, stehen in zukünftigen Wohnumgebungen auch Kostenminimierung und Komfort im Vordergrund, welches durch den Einsatz von Technik erreicht werden kann, vgl. Capgemini Consulting (2011).
Gegenüber dem Menschen steht die Technik in einer Wohnumgebung. Hiermit ist das was aus der Frage gemeint, denn die Informationen und Funktionen einer Wohnumgebung dürfen nicht für alle Geräte dieser Umgebung zur Verfügung stehen. So könnte einem Fernseher beispielsweise Zugriff auf die Lichtsteuerung gewährt werden, um die Lichtsituation der Stimmung des Films anzupassen. Zugriff auf diverse andere Funktionen oder Informationen, wie beispielsweise der Kühlschrankinhalt, Lieblingsmusik der Bewohner, etc. werden für den Betrieb eines Fernsehers nicht benötigt.
Für die Zukunft muss noch ein gesundes Maß zwischen denn gesammelten und bereitgestellten Informationen und den damit bereitgestellten Funktionen ermittelt werden. Die Informationen sollen zwar dazu dienen dem Menschen mehr Komfort und Sicherheit zu bieten, jedoch können sie dem Menschen auch schaden. Dies könnte der Fall sein, wenn die Geräte die gesammelten Informationen an Dritte, wie beispielsweise Hersteller oder Behörden, weitergeben.

Neben Wohnumgebungen können Berechtigungsstrukturen auch in vergleichbaren Umgebun-

gen, wie Flugzeugen oder dem Automobil eingesetzt werden. In diesen Umgebungen hat die Anzahl technischer Systeme ebenfalls stark zugenommen. Da diese immer weiter miteinander vernetzt werden, bedarf es auch in diesen Umgebungen Berechtigungsstrukturen, welche den Zugriff klar regeln. Ein Beispiel für die Zusammenführung verschiedener Systeme ist das Projekt *PRORETA 3*, welches die Ausnutzung vorhandener Sensorinformationen maximieren möchte. Ziel ist die Entwicklung eines unfallvermeidenden Fahrzeugs durch die Fusion der einzelnen Assistenzsysteme eines Wagens, vgl. Technische Universität Darmstadt.

1.2. Ziele der Arbeit

Ziel dieser Arbeit ist die Entwicklung eines kontextsensitiven Berechtigungssystems, welches den Zugriff auf Ressourcen in einer Wohnumgebung regulieren kann. Dabei sollen nicht nur statische, sondern auch dynamische Berechtigungen realisierbar sein, welche in Abhängigkeit zum aktuellen Kontext stehen. Die Integration von dynamischen Berechtigungen ermöglicht die Abbildung jener Zugriffskontrollen, die durch statische Berechtigungen nicht realisierbar sind. So können beispielsweise Berechtigungen anhand der aktuellen Position oder dem Zustand einer Person definiert werden. Dies ermöglicht die Abbildung alltäglicher Situationen eines Menschen. Des Weiteren wird durch die Verwendung von dynamischen Berechtigungen die Sicherheit des Systems gesteigert.
Ein weiterer wichtiger Punkt ist die Zuweisung von Berechtigungen auf einzelne Funktionen von Geräten und Ressourcen. Wäre nur eine generelle Zugriffskontrolle auf Ressourcen einer Wohnumgebung möglich, könnten alltägliche Situationen nicht durch das Berechtigungssystem abgebildet werden. Verdeutlicht werden soll dies anhand der Steuerung eines TV-Gerätes. Mit einer generellen Zugriffskontrolle könnte nur definiert werden, ob die Person den TV verwenden darf oder nicht. Darf das Gerät verwendet werden, besitzt die Person auch die Berechtigung alle weiteren Funktionen des TV zu nutzen, wie beispielsweise Aufnahmen zu programmieren oder aufgenommene Sendungen anzuschauen, jeden Kanal einzuschalten oder auf jede Applikation zuzugreifen, die auf dem Fernseher zur Verfügung gestellt wird. Durch das Setzen von Berechtigungen auf einzelne Funktionen des Fernsehers kann dies verhindert werden.
Neben den Funktionalitäten des Berechtigungssystems soll dieses möglichst modular aufgebaut sein. Hierdurch soll gewährleistet werden, dass dieses System nicht nur im Living Place Hamburg Verwendung findet, sondern auch in andere Smart Homes integriert werden kann. Dafür soll das System in einzelne Teilkomponenten gegliedert werden. So können die Komponenten einfach an neue Umgebungen angepasst oder sogar ausgetauscht werden.

1.3. Gliederung

Diese Arbeit ist in sechs Kapitel unterteilt. Das erste Kapitel dient der Einführung in die Thematik der Arbeit. Dafür werden die Ziele der Arbeit definiert und die Gliederung vorgestellt, vgl. Kapitel 1.

Im Anschluss werden in Kapitel 2 die erforderlichen Grundlagen zu dieser Arbeit aufgeführt. Hierzu zählt zum einen die Projektumgebung 2.1, in der diese Arbeit durchgeführt wird. Zum anderen weitere Grundlagen, wie Berechtigungskonzepte 2.2, die dem Verständnis der Arbeit dienen.

Darauffolgend werden in der Analyse 3 die zuvor aufgestellten Ziele der Arbeit analysiert. Dafür werden in Abschnitt 3.1 zunächst vier Szenarien vorgestellt. Anhand derer mögliche Problemstellung identifiziert werden. Diesbezüglich werden Informationen aus den Szenarien extrahiert und kategorisiert, um mittels dieser Informationen den vorherrschenden Kontext zu identifizieren, vgl. Abschnitt 3.2. Des Weiteren wird im Abschnitt 3.3 die vorhandene Umgebung mit ihren Sensoren, Aktoren und Services vorgestellt. Daraufhin werden vergleichbare Arbeiten aufgeführt, die sich im gleichen Kontext aufhalten, wie diese Arbeit, vgl. Abschnitt 3.4. Das Kapitel schließt mit den Abgrenzungen 3.5 und einem anschließenden Fazit 3.6.

In Kapitel 4 wird auf Basis der zuvor gewonnenen Informationen ein Design und die anschließende Realisierung vorgestellt. Dafür wird zunächst eine Überblick der gewünschten Funktionalitäten gegeben. Des Weiteren werden Arbeiten vorgestellt, die als Grundlage dieser Arbeit dienen, vgl. Abschnitt 4.1. Nachfolgend wird in Abschnitt 4.2 der gewählte Verzeichnisdienst und deren Aufbau dargestellt. Als Nächstes wird in den Abschnitten 4.3 und 4.4 beschrieben, welches System zur Erfassung und Verarbeitung der Kontextinformationen verwendet wird und welche Möglichkeiten bestehen, auf diese zu reagieren. Anschließend wird die Realisierung der Konzepte beschrieben, vgl. Abschnitt 4.5. Mit dem darauffolgenden Fazit 4.6 wird das Desingkapitel beendet.

Das vorletzte Kapitel beschäftigt sich mit der Integration der realisierten Komponenten in das Living Place Hamburg. Es wird beschrieben, auf welchem System die Komponenten laufen und wie das Gesamtsystem konfiguriert ist, vgl. Kapitel 5.

Das letzte Kapitel beginnt mit einer Zusammenfassung der Arbeit 6.1. Abschließend wird im

Abschnitt 6.2 auf mögliche Konsequenzen dieses Ansatzes eingegangen. Des Weiteren werden zukünftige Erweiterungen und Einsatzmöglichkeiten aufgezeigt.

2. Grundlagen

2.1. Living Place Hamburg

Das Living Place Hamburg entstand im Jahr 2009 als interdisziplinäre Laborumgebung in der Studenten verschiedener Fachbereiche die Auswirkungen neuer Technologien in einer Wohnumgebung erforschen können. Dabei werden Fragen aus dem technischen Umfeld sowie Fragen des modernen urbanen Lebens geklärt. Dazu zählen neue Formen des sozialen Lebens, wie beispielsweise eine Neugestaltung der Nachbarschaftsbeziehungen durch technische Unterstützung oder die Verschmelzung zwischen Arbeit und Freizeit, vgl. von Luck u. a. (2010).

Abbildung 2.1.: Bestandsgebäude der HAW

Das Living Place Hamburg wurde in ein Bestandsgebäude der Hochschule für Angewandte Wissenschaften Hamburg integriert (siehe Abbildung 2.1). Insgesamt umfasst das Living Place eine Fläche von 140 m^2 und ist in verschiedene Bereiche unterteilt. Die entwickelten Projekte der Studenten werden in den *Living area* Bereich integriert. Dieser Bereich ist im Stil eines Lofts aufgebaut, welches in verschiedene Wohnbereiche unterteilt ist. Dazu gehören ein Ess-, Schlaf- und Wohnbereich sowie eine voll ausgestattete Küche. Daneben befindet sich der *Control room*. In ihm können über diverse Monitore die Vorgänge im Living Place kontrolliert und überwacht werden. Das System kommt vor allem bei Machbarkeitsstudien zum Einsatz, in denen die Studenten die Ansätze ihrer Arbeiten evaluieren. Des Weiteren existieren noch die *Development area* und eine Werkstatt. Ersteres besteht aus insgesamt zwei Büroräumen, in denen die Studenten Softwareentwicklung betreiben. Die Werkstatt befindet sich links neben dem Kontrollraum und dient den Studenten zur Herstellung von Hardwarekomponenten. Der

geschilderte Aufbau ist in der Abbildung 2.2 dargestellt.

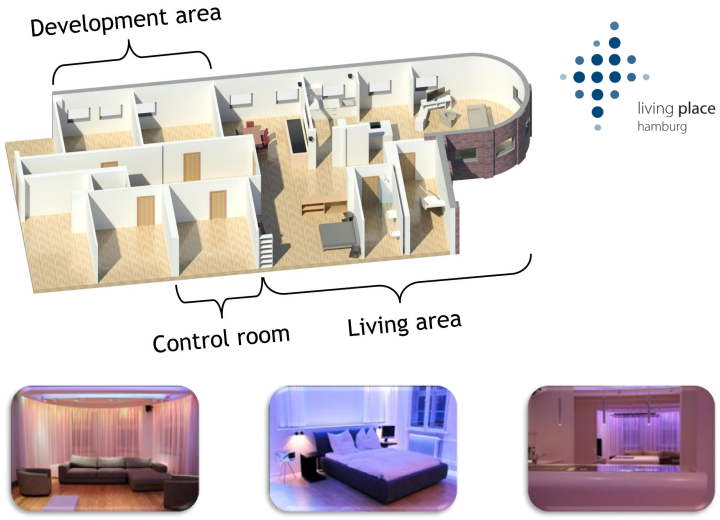

Abbildung 2.2.: Aufteilung des Living Place Hamburg (Quelle: Voskuhl (2011))

Die Verwaltung des Living Place Hamburg liegt in der Hand der Professoren von Luck, Klemke und Wendholt, unter deren Anleitung Projekte für Studenten entstehen. Seit der Eröffnung des Living Place sind eine Vielzahl von Projekten entstanden, deren Dokumentation in einem projektbezogenen Wiki zu finden ist. Um die Wiederverwendbarkeit zu steigern und die Integration verschiedener Bibliotheken anderer Studenten in das eigene Projekt zu verbessern, wurde im Rahmen der Masterarbeit von Kjell Otto ein Maven sowie ein Git Repository erstellt, vgl. Otto (2013).
Die Infrastruktur des Living Place Hamburg ist ausgestattet mit einem Message Broker System. Die Kommunikation wurde absichtlich sehr simple gehalten, um den Studenten einen möglichst leichten und schnellen Einstieg in die Umgebung zu ermöglichen. Somit ist das Abfragen von Sensoren oder die Ansteuerung von Aktoren in wenigen Schritten möglich. Eine Auflistung dieser ist in 3.3 zu finden.

Da sich die Projekte im Living Place Hamburg im Kontext von Ubiquitous Computing, Smart Homes und Context awareness aufhalten, werden diese Begriffe im Folgenden erläutert.

2.1.1. Ubiquitous Computing

Ubiquitous Computing beschreibt die Vision einer Zukunft, in der die heutigen Personal Computer durch sogenannte intelligente Gegenstände ersetzt wurden. Die Idee dahinter ist, dass die Gegenstände den Menschen unterstützen, ohne das diese ihn ablenken. Dabei werden sie soweit in die Umgebung des Menschen integriert, dass dieser die Existenz der rechnergestützten Systeme gar nicht mehr wahrnimmt. Der Begriff wurde von Mark Weiser in seinem Artikel *The Computer for the 21st Century* geprägt, vgl. Weiser (1991).
In dem Buch *Ubiquitous Computing Fundamentals* beschrieb John Krumm drei elementare Zeitabschnitte der Rechenleistung, vgl. Krumm (2009). Es begann alles mit der Mainframe Ära. Darauf folgte das Zeitalter des Personal Computers. Die dritte und letzte Ära ist die des Ubiquitous Computing. Durch anhaltende Entwicklungen im Bereich der Rechenleistung und Mikroelektronik können immer kleinere Gegenstände mit Rechen- oder Rechnerleistung ausgestattet werden. Schon jetzt umgeben uns täglich eine Vielzahl von elektrischen Gegenständen, wie Kreditkarten, Haushaltsgeräte, Smartphones oder Autos. Dies zeigt, wir befinden uns bereits in den Anfängen des Ubiquitous Computing.
Das Büro für Technologiefolgen-Abschätzungen beim Deutschen Bundestag veröffentlichte im Jahr 2009 einen Zukunftsreport zum Thema Ubiquitous Computing. Unter anderem wurden in diesem Bericht auch die folgenden technischen Merkmale identifiziert, vgl. (Friedewald u. a., 2009, S. 41f.):

- Dezentralität bzw. Modularität
- Einbettung
- Mobilität
- (Spontane) Vernetzung
- Kontextsensitivität
- Autonomie
- Autarkie

2.1.2. Smart Homes

Smart Homes sind unter der Kategorie Ubiquitous Computing einzuordnen. Unter diesem Begriff versteht man eine intelligente Wohnumgebung, welche mittels Sensoren, Aktoren und der Analyse von Daten bestimmte Dienste bereitstellen kann. Anhand der Sensorik

können Informationen aus der Wohnumgebung gewonnen werden. Diese werden analysiert, um Rückschlüsse auf die momentane Situation zu gewinnen und gegebenenfalls mittels der vorhandenen Aktorik einzugreifen. Ziel ist es, dem Menschen alltäglich Aufgaben abzunehmen oder zu erleichtern und eventuelle Gefahrensituationen zu erkennen oder gar zu vermeiden. Augusto identifizierte drei Merkmale eines Smart Homes, vgl. Augusto (2007):

- Erhöhte Sicherheit
  Durch Erkennung gefährlicher Umstände
- Komfort
  Anhand von Regulierungen, beispielsweise Raumtemperatur oder Lichtstimmungen
- Wirtschaftlichkeit
  Einsparpotential durch die Regelung der Systeme

2.1.3. Context awareness

Context awareness beschreibt die Möglichkeit einer Anwendung auf eine vorherrschende Situation zu reagieren. Ein System, welches diese Fähigkeiten besitzt, benötigt Zugriff auf die Umgebung, um Informationen zu erhalten. Auf Basis dieser Informationen kann anschließend durch die Ansteuerung von Aktoren ein reaktionäres Verhalten erzeugt werden. Des Weiteren benötigt die Anwendung Möglichkeiten zur Verarbeitung und Analyse der Daten, vgl. Schilit u. a. (1994).
Zur Kontextgewinnung müssen äußere Einflüsse ebenso herangezogen werden, wie vorhandene Kommunikationsinformationen zwischen Menschen und Computer. Eine generelle Definition von Context awareness wurden von Dey und Abowd im Artikel *Towards a Better Understanding of Context and Context-Awareness* gewählt:

> *A System is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task.*
> *Anind K. Dey and Gregory D. Abowd*
> *Abowd u. a. (1999)*

Andere Definitionen, wie beispielsweise von Hull u. a. (1997), von Context Aware Systemen beinhalten die Erkennung, Interpretation und eine anschließende Reaktion in der Umgebung. Diese Präzisierung schließt jedoch jene Systeme aus, die den Kontext einer Umgebung bestimmen, aber nicht auf ihn reagieren. Durch die generalisierte Definition von Abowd u. a. können auch Systeme welche nicht auf eine Situation reagieren als Context awareness bezeichnet werden.

2.2. Berechtigungskonzepte

Berechtigungskonzepte werden verwendet, um Ressourcen in einem System vor dem Zugriff unbefugter Personen zu schützen. Im Laufe der Zeit sind viele verschiedene Konzepte zu diesem Thema entstanden. Drei der meist verwendeten Konzepte werden im Folgenden beschrieben.

2.2.1. Mandatory Access Control

Das Mandatory Access Control Modell nutzt verschiedene Techniken zum Schutz sensibler Ressourcen. Dabei wird der Zugriff auf die Daten nicht nur anhand der Zuordnung einer Benutzeridentität zu einer Ressource geregelt, sondern es werden sogenannte Schutzstufen verwendet. Diese Schutzstufen kann man sich als vertikale Schichten vorstellen, wie die Stockwerke eines Gebäudes. Sind Personen und Daten einer vertikalen Schicht zugeteilt, haben sie Zugriff auf alle Daten dieser und der darunter liegenden Schichten. In Verbindung mit dem Gebäudebeispiel würde das bedeuten, besitzt eine Person die Berechtigung das 10. Stockwerk zu betreten, darf es auch die darunterliegenden betreten. Allerdings besteht keine Zutrittsberechtigung für die darüber liegenden Stockwerke. Die Einführung solcher Schichten bietet einen zusätzlichen Schutz sensibler Daten und findet daher auch Anwendung im militärischen Umfeld, bei Behörden oder in der Gesundheitsbranche. Am Beispiel des militärischen Umfeldes könnten die Schichten möglicherweise in Streng Geheim, Geheim, Vertraulich und Öffentlich aufgeteilt sein. Im Laufe der Zeit wurden dem Multi-Level-Sicherheitssystem weitere Modelle hinzugefügt. Unter anderem das Bell LaPadula, Biba oder LoMAC. Diese Modelle erweitern die Struktur des Mandatory Access Control Modells um einige Eigenschaften, wie die Sicherstellung der Integrität der Daten oder dessen Überführung in andere Schutzstufen, vgl. Sandhu (1993).
Multiliterale Sicherheitsmodelle bilden im Gegensatz zu den Multi-Level-Sicherheitssystemen einen komplexeren Aufbau mittels sogenannter Segmente. Durch die Kombination mehrerer Schutzstufen mit Codewörtern werden den vertikalen Schichten zusätzlich horizontale Schichten hinzugefügt. Erweiterungen dieses Modells stellen das Compartment- oder Lattice-Modell und die Chinese Wall dar, vgl. Sandhu (1993).

2.2.2. Discretionary Access Control

Das Discretionary Access Control Modell bildet den Zugriff auf eine Ressource allein auf Basis der Benutzeridentität ab. Dies bedeutet, für jeden Benutzer muss einzeln festgelegt werden, worauf und wie er Zugriff hat. Am Beispiel eines Zugriffs auf eine Datei heißt das, der Name beschreibt, worauf der Benutzer Zugriff hat, während das Zugriffsrecht beschreibt, wie der

Zugriff auf die Datei besteht. Zum Beispiel das Recht die Datei zu lesen, zu schrieben, etc. Den Grundstein dieses Berechtigungsmodells legten Harrison u. a. (1976) im Jahr 1976.
Da diese Berechtigungen für jeden Benutzer angelegt werden müssen, ist der Aufwand für die Erstellung und Wartung des Systems in Umgebungen mit vielen Personen und Ressourcen sehr hoch.

2.2.3. Role Based Access Control

Das zuvor beschriebene Discretionary Access Control Modell wird durch das Role Based Access Control Modell um einige Eigenschaften erweitert. Dazu zählen die Verwendungen von Benutzergruppen und -rollen. Durch die Benutzergruppen können die Benutzer hinsichtlich organisatorischer Strukturen gruppiert werden. Des Weiteren wird die Zuweisung von Berechtigungen erleichtert, da die Zuweisung nicht mehr auf einzelne Benutzer erfolgen muss, sondern über die Gruppe realisiert werden kann. Berechtigungen werden hingegen durch Rollen gruppiert. Sie beschreiben, welche Rechte für die Ausübung eines bestimmten Arbeitsprozesses benötigt werden. Dies erleichtert zum einen die Zuweisung von Rechten, zum anderen erlaubt es die dynamische Erweiterung aktueller Berechtigung durch Zuweisung einer Rolle.

Im Jahr 1996 wurde der RBAC-Standard veröffentlicht, vgl. Sandhu u. a. (1996). Seit seiner Veröffentlichung wurde dieser Standard von der Version 0 bis zur Version 3 weiterentwickelt. Die Abbildung 2.3 stellt die Beziehungen zwischen den einzelnen Versionen dar. Die Version

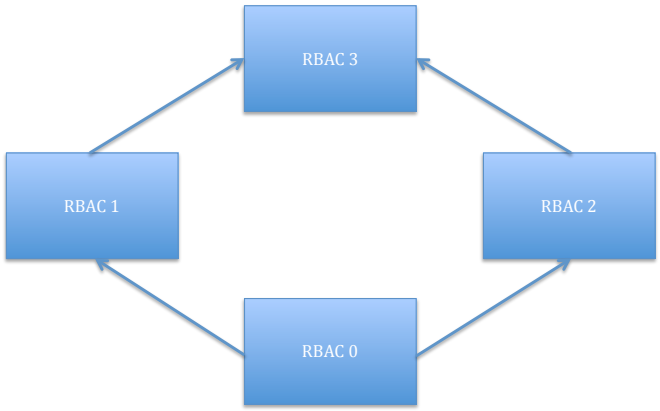

Abbildung 2.3.: Beziehung zwischen den RBAC-Versionen

0 wird auch als Kernmodell bezeichnet, da in ihr die grundlegenden Konzepte des Modells

enthalten sind. Hierzu zählen die Benutzer, Rollen, Rechte und Sitzungen. Die Sitzungen dienen der dynamischen Aktivierung von Rechten. Dies bewirkt, dass der Anwender die Rechte genau zu dem Zeitpunkt besitzt, wenn er sie wirklich benötigt.
Die Version 1 des RBAC Standards erweiterte das Modell um eine Rollenhierarchie. Somit können die Rechte einer Rolle an darunterliegenden Rollen vererbt werden.
In der Version 2 wurde das Konzept zur Einschränkung von Rollen definiert. Dies verhindert eine Zuweisung konfliktbehafteter Rollen zu einem Benutzer.
Die Zusammenführung der Erweiterungen aus der Version 1 und 2 erfolgte in der Version 3. Darüber hinaus wurden in dieser Version keine weiteren Neuerungen eingearbeitet. Tiefer gehende Informationen zu Rollen- und Berechtigungskonzepten sind in Tsolkas und Schmidt (2010) zu finden.

2.3. OSGi-Plattform

Die OSGi Alliance (früher Open Services Gateway initiative) ist eine Plattform, welche es auf der Java Virtual Machine (JVM) ermöglicht, eine offene und einheitliche Entwicklung von Diensten zu realisieren. Hierbei steht die Bereitstellung und Verwaltung von Diensten im Vordergrund. Die Kernkompetenz der Plattform besteht in dem Konzept der Modularisierung. Dies bedeutet, es können verschiedene Softwarekomponenten dynamisch zur Laufzeit zur OSGi-Plattform hinzugefügt oder entfernt, gestartet oder gestoppt werden. In der Terminologie der Plattform werden die Softwarekomponenten als Bundles bezeichnet, vgl. Weber u. a. (2010).
Seit April 2011 steht die der aktuelle Version 4.3 der OSGi-Plattform zur Verfügung. Im Juni 2012 wurde die Version 5 der Plattform vorgestellt, welche sich zurzeit in der Entwicklungsphase befindet, vgl. OSGi Alliance.

2.3.1. OSGi-Schichtenmodell

Das OSGi-Framework ist eine Laufzeitumgebung auf Basis der Java Virtual Machine. Die Funktionalitäten die dieses Framework bietet, werden anhand verschiedener Schichten ermöglicht. Der Schichtenaufbau ist in der Abbildung 2.4 dargestellt. Die Sicherheitsschicht ist vertikal zu den übrigen Schichten angeordnet, da sie Sicherheitsmechanismen für das gesamte Framework zur Verfügung stellt. Im Folgenden wird ein grober Überblick über die einzelnen Segmente des Frameworks gegeben. Eine detailliertere Beschreibung der Schichten würde den Rahmen dieser Arbeit verlassen. Werden tiefer gehende Informationen benötigt sind diese in den OSGi-Spezifikation zu finden, vgl. OSGi Alliance (2011). Einen guten Überblick gibt ebenfalls das Buch *OSGi für Praktiker*, vgl. Weber u. a. (2010).

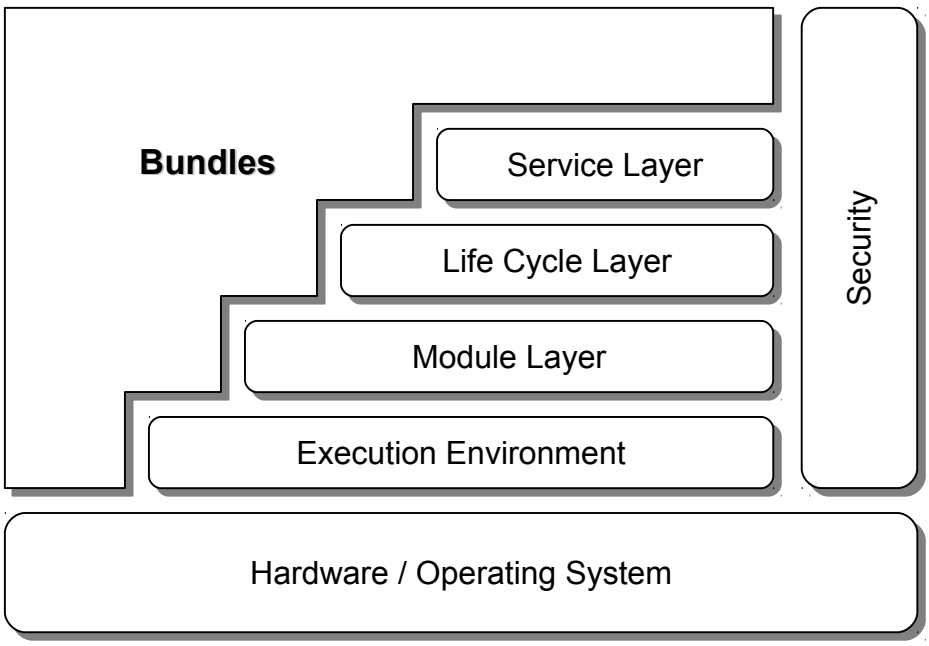

Abbildung 2.4.: OSGi Schichtenmodell (Quelle: (Otto, 2013, S. 44))

Security: Der Security Layer ist optional. Dies ist gerade für Plattformen mit beschränken Ressourcen von Vorteil. Im Inneren der OSGi Sicherungsschicht wird die Java-2-Sicherheitsarchitektur verwendet.
Anhand der Sicherungsschicht kann das Laden und Ausführen von Klassen zur Laufzeit erlaubt werden. Des Weiteren ermöglicht die Schicht das Ausüben von Sicherheitsprüfungen und die Signierung von Bundles. Dadurch kann sichergestellt werden, dass seit der Signierung, der Inhalt des Bundles unverändert ist, vgl. (OSGi Alliance, 2011, S. 11f.)

Service Layer: Der Service Layer definiert ein dynamisches Modell und kollaboriert eng mit dem Life Cycle Layer, da diese Schicht die vorhandenen Services verwaltet. Services bestehen aus normalen Java-Objekten und -Interfaces. Der Service kann bei der Service-Registry über ein oder mehreren Interfaces registriert werden. Dabei ist zu beachten, dass ein Service innerhalb eines Bundles läuft. Die Registrierung wird über die *registerService*-Methode des Bundle-Context-Objekts ausgeführt. Die Referenz auf den Service wird dann in der OSGi Registry registriert und kann von anderen Bundles wieder abgerufen werden. Das Arbeiten mit Referenzen kann jedoch zu den sogenannten Stale Reference führen. Das sind Referenzen, die zur Zeit des Aufrufs nicht mehr verfügbar sind, weil

beispielsweise das Bundle, in dem die Klasse vorhanden ist, gestoppt wurde. Um dieses Problem zu umgehen, wird die Nutzung der Services über den Service Tracker oder Declarative Service empfohlen. Des Weiteren besteht für Bundles die Möglichkeit sich über den Zustand eines Services über den *ServiceListener* informieren zu lassen, vgl. (OSGi Alliance, 2011, S. 111f.)

Life Cycle Layer: Der Life Cycle Layer ist, wie der Name schon sagt, für den Lebenszyklus eines Bundles verantwortlich. Die Steuerung des Bundles erfolge über die Implementierung des *BundleActivator*-Interface. Über dessen Methoden erhält das OSGi-Framework die Möglichkeiten zum Starten und Stoppen des Bundles. Welche Klasse das BundleActivator-Interface implementiert, wird in der Manifest-Datei festgehalten. Dadurch wird dem Framework der BundleActivator bekannt gemacht. Durch diese Methoden erhält das Bundle auch Objekte vom Typ *BundleContext*, wodurch der Zugriff auf Informationen des Frameworks hergestellt werden kann.
Ein Bundle, welches sich in der OSGi-Plattform befindet, kann verschiedene Stadien annehmen. Diese sechs Zustände und Zustandsübergänge sind in Abbildung 2.5 dargestellt, vgl. (OSGi Alliance, 2011, S. 79f.).

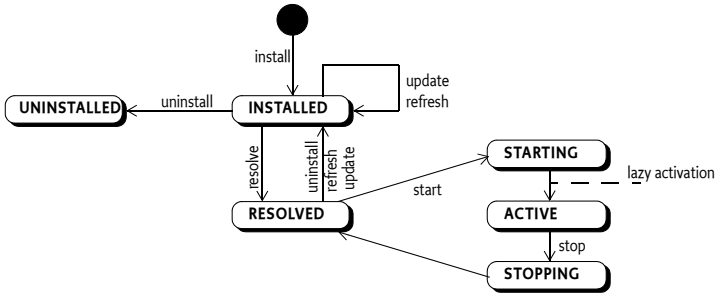

Abbildung 2.5.: Zustände und Zustandsübergänge von Bundles (Quelle: (OSGi Alliance, 2011, S. 90).)

Module Layer: Der Module Layer erweitert das Modulkonzept der Standard-Java-Plattform, da diese nur eine eingeschränkte Unterstützung bietet. Diese Schicht stellt *ClassLoader* zur Verfügung, welche das Laden unterschiedlicher Bundles ermöglicht. Jedes Bundle besitzt hierfür seinen eigenen *ClassLoader*, der unter anderem auch für den Prozess des Resolving verantwortlich ist. Darunter versteht man das Auflösen aller Abhängigkeiten, die beispielsweise in der Manifest-Datei definiert wurden. Die anschließende Verknüpfung der Packages mit dem Bundle gehört ebenfalls zu diesem Prozess, vgl. (OSGi Alliance, 2011, S. 25 ff)

Execution Environment: Das Execution Environment repräsentiert die Laufzeitumgebung. Müssen Bundles oder Services in einer speziellen Laufzeit ausgeführt werden, können diese in der Manifest-Datei mit dem Eintrag *Bundle-RequiredExecutionEnvironment* festgelegt werden, vgl. (OSGi Alliance, 2011, S. 35f.)

2.3.2. Bundles

Bundles können aus einer Menge von Java-Klassen und anderen verschiedenen Ressourcen, wie Bildern, HTML-, XML- Dateien, etc. bestehen. Die Bundles können unabhängig gestartet und gestoppt werden. Ferner kann der Zustand der Komponenten überwacht werden, womit sowohl der gesamte Lebenszyklus, als auch die Versionsverwaltung abgebildet werden können, vgl. Weber u. a. (2010).
Die Verbindung zwischen Java und OSGi wird über sogenannte Manifest-Dateien hergestellt. In ihr werden alle notwendigen Metainformationen zur Spezifizierung des Bundles definiert. Um das Bundle in einer OSGi-Umgebung hinzuzufügen, muss dieses kompiliert werden. Hierdurch einsteht eine Jar-Datei, welche die Manifest-Datei beinhaltet und somit der OSGi-Umgebung die benötigten Informationen bereitstellt.
Ein auf OSGi basierendes Softwaresystem besteht aus einer Vielzahl von Bundles. Um den korrekten Ablauf des Systems zu gewährleisten, müssen die Bundles selbstverständlich untereinander kommunizieren. Hierfür bietet OSGi die Möglichkeit in der Manifest-Datei einzutragen, welche Packages exportiert und importiert werden sollen. Packages, die exportiert werden, dienen der Bekanntmachung. Dies bedeutet, dass andere Bundles Zugriff auf die Funktionalitäten dieses Bundles erhalten. Packages die importiert werden, dienen der Verwendung innerhalb des Bundles. Dies bietet unter anderem den Vorteil, dass über das Importieren und Exportieren von Packages Interfaces klar definiert werden können. Somit ist es möglich, die Implementierung zu verstecken und die Funktionalitäten nur über die Schnittstellen bereitzustellen, vgl. Fildebrandt (2012).

2.3.3. Services

Services sind genau wie Bundles ein wichtiger Teil der OSGi Struktur. Da sie über die Bundle-Grenzen hinweg erreichbar sind, werden sie für die Kommunikation zwischen Bundles eingesetzt. Die Verwendung von Services ist die OSGi konforme Art der Programmierung, welche erst die lose Kopplung zwischen den Bundles ermöglicht. Ohne Services würden die Bundles sich untereinander referenzieren, was zu Abhängigkeiten führt und somit das Konzept der Modularisierung aufhebt.

Um Services für alle Bundles verfügbar zu machen, werden sie in der OSGi Registry abgelegt. Diese beinhaltet eine Map über String-Schlüssen und Objektinstanzen als dazugehörige Werte. Wobei der String den vollständigen Namen des Interfaces repräsentiert und die Objektinstanz die konkrete Implementierung beherbergt, vgl. Fildebrandt (2012).

3. Analyse

Diese Arbeit beschäftigt sich mit der Entwicklung einer Berechtigungsstruktur für Smart Homes, welche den Zugriff auf Ressourcen und deren Funktionen, in Abhängigkeit von Kontextinformationen, reglementieren soll. Dafür werden in der Analyse zunächst einige Szenarien 3.1 vorgestellt. Anschließend werden die Informationen zur Erkennung des Kontextes extrahiert 3.2.1 und kategorisiert 3.2.2.
Darauffolgend wird die Umgebung bezüglich ihrer Sensorik, Aktorik, Services und Kommunikationsschnittstelle vorgestellt. In Bezug auf die zu wählende Datenhaltung, der Gewinnung von Kontextinformationen und der Informationsweitergabe, sind diese Informationen Voraussetzungen für das Design der Berechtigungsstruktur.
Anschließend werden verwandte Arbeiten vorgestellt und bewertet 3.4. Des Weiteren werden Abgrenzungen 3.5 vorgenommen und ein Fazit 3.6 der Analyse wird präsentiert.

3.1. Szenarien

In diesem Abschnitt werden vier Szenarien vorgestellt, welche einen Überblick über die aktuellen Problemstellungen geben sollen. In den Szenarien werden fünf fiktive Charaktere vorkommen. Das Ehepaar Alice und Bob mit Ihrer Tochter Carol, Oma June und der Freund der Familie James.
Anhand dieser Szenarien werden im Anschluss Problemstellung identifiziert und Anforderungen an das System aufgestellt.

3.1.1. Szenario: Familienurlaub

Der Familie macht eine vierwöchige Rundreise durch die USA. Für die Zeit haben Sie James gebeten, sich um die Wohnung zu kümmern. In der intelligenten Wohnung der Familie wird die Wohnungstür aber schon längst nicht mehr über einen Schlüssel, sondern über die Smartphones der Familienmitglieder geöffnet. Aus diesem Grund wurde im Berechtigungssystem der intelligenten Wohnung hinterlegt, dass James Zutritt zur Wohnung, mittels seines Smartphones, erhält, sobald die Familie im Urlaub ist.

Über die in den Töpfen eingesetzten Feuchtigkeitssensoren (vgl. Koubachi AG) besitzt die intelligente Wohnung einen Überblick über den aktuellen Zustand der Pflanzen. Sobald einige Pflanzen drohen auszutrocknen, wird James darüber informiert. Somit braucht James nicht auf Verdacht zur Wohnung der Familie fahren, sondern nur, wenn es unbedingt notwendig ist.
Die Familie ist nun schon drei Tage im Urlaub und die intelligente Wohnung hat James gemeldet, dass über 50% der Pflanzen Wasser benötigen. James kommt deshalb vorbei, um die Blumen in der Wohnung zu gießen. Wenn er vor der Wohnungstür steht, berührt er mit seinem Smartphone ein Lesegerät neben der Tür. Über eine Passworteingabe authentifiziert James sich am Berechtigungssystem, welches anschließend die aktuellen Berechtigungen überprüft. Das System erkennt, dass sich die Familie im Urlaub befindet, James vor der Wohnungstür steht und er die Berechtigung zum Öffnen der Wohnungstür besitzt. Somit sind alle Voraussetzungen erfüllt und die Wohnungstür wird geöffnet. Bob bekommt nun eine Mitteilung auf seinem Smartphone, die ihn über das Betreten und Verlassen der Wohnung informiert.

3.1.2. Szenario: Wie lange und welche Inhalte darf Carol im Fernsehen schauen

Carol ist ein Fernsehjunkie. Um diese Gewohnheiten etwas zu verändern, haben Alice und Bob beschlossen, das Fernsehkontingent von Carol einzuschränken. Dafür haben sie im Berechtigungssystem einige Einstellungen festgelegt. Hierzu gehört welche TV-Sendungen Carol nicht mehr sehen darf, wie lange sie insgesamt Fernsehen darf und zu welchen Zeiten.
Die TV-Services der Wohnung verfolgen nun genau Wann, Wo, Was und mit Wem Carol Fernsehen schaut. Wird eine der Regeln verletzt, wird der Fernseher nach einer kurzen Meldung für Carol abgeschaltet oder gar nicht erst eingeschaltet. Anders sieht es aus, wenn Carol nicht alleine, sondern mit einem Elternteil vor dem Fernseher sitzt. Ist dies der Fall wird eine Nachricht anzeigt, welche den Elternteil über die aktuelle Situation informiert und Vorschläge für das weitere Vorgehen unterbreitet.

3.1.3. Szenario: Oma June hat einen Kreislaufzusammenbruch

Es ist am frühen Vormittag und die ganze Familie ist aus dem Haus, bis auf Oma June. Wie jeden Morgen sitzt sie auf dem Sofa und schaut ihre Lieblingssendung. Sie schaltet den Fernseher auf Time Shift, um in die Küche zu gehen und sich eine Tasse Tee zu kochen. Auf dem Weg dort hin stolpert June und bleibt bewusstlos auf dem Boden im Esszimmer liegen. Die intelligente Wohnumgebung bemerkt, dass sich June schon seit einer halben Minute nicht mehr bewegt hat und das in einem Bereich, in dem es dafür keinen ersichtlichen Grund gibt.

Denn hier stehen keine Möbel, Haushaltsgeräte oder Ähnliches an dem June länger verweilen könnte. Daraufhin versucht die intelligente Wohnumgebung, durch Licht- und Tonsignale eine Reaktion bei June hervorzurufen. Nachdem keine Reaktion erkennbar war, wird die Hausnotrufzentrale verständigt (vgl. DRK). Der Mitarbeiter der Notrufzentrale versucht mit June Kontakt aufzunehmen, jedoch ohne Erfolg, woraufhin der Rettungsdienst und die Familienmitglieder verständigt werden. Um eine schnellere erste Hilfe zu gewährleisten, werden zusätzlich die Nachbarn verständigt. Sobald jemand der verständigten Personen vor der Tür steht, erkennt dies die intelligente Wohnung und öffnet diese. Durch Lichtsignale werden die Personen direkt zu June ins Esszimmer geleitet. Nach 10 Minuten treffen die Rettungskräfte ein und übernehmen die Versorgung von June, welches bis dato ein Nachbar übernommen hatte. Aufgrund der schnellen Hilfe konnte June gerettet werden.

3.1.4. Szenario: Parkplatz-Sharing

Die Familie besitzt einen Parkplatz in der Tiefgarage ihres Wohnhauses. Da dieser Parkplatz meistens den ganzen Tag frei steht, entscheidet sich Bob zu einer Anmeldung bei einem Parkplatz-Sharing Anbieter. Somit kann der Parkplatz auch am Tage von anderen Personen benutzt werden und die Familie kann noch etwas Geld dazu verdienen. Sensoren im Boden des Parkplatzes melden der intelligenten Wohnung, ob ein Fahrzeug gerade den Parkplatz belegt. Fährt Bob von zu Hause weg meldet er dem Anbieter, wie lange der Parkplatz voraussichtlich frei ist. Der Anbieter gibt diese Daten für seine Mitglieder frei, woraufhin jemand den Platz mietet. Steht der Mieter vor dem Eingang der Tiefgarage, sendet dieses durch eine Applikation auf seinem Smartphone den Befehl zum Öffnen des Tores. Der Anbieter überprüft, ob diese Person berechtigt ist, das Tor zu öffnen und gibt dies anschließend an die intelligente Wohnung weiter. Diese veranlasst nun die Öffnung des Garagentors und leitet den Mieter durch im Boden eingelassen Signallampen zum richtigen Parkplatz. Die Wohnung meldet dem Parkplatz-Sharing Anbieter, dass der Platz nun besetzt ist, worauf der Anbieter den Mietvorschlag von seiner Plattform entfernt, bis der Platz wieder frei wird.

3.2. Kontextanalyse

Die im vorherigen Abschnitt 3.1 vorgestellten Szenarien geben Aufschluss darüber, inwiefern vorher definierte Berechtigungen in bestimmten Situationen angepasst werden können. Darüber hinaus wird ersichtlich, welche Informationen in den jeweiligen Szenarien vorhanden sein müssen, um den Kontext zu erkennen.
Im Folgenden wird zunächst der Begriff Kontext 3.2.1 und die verschiedenen Arten von In-

formationen erläutert. Dies ist eine Voraussetzung um die anschließende Kategorisierung der Informationen 3.2.2 aus den Szenarien nachzuvollziehen. Darauffolgend wird die Verarbeitung der Informationen aufgezeigt 3.2.3.

3.2.1. Kontextinformationen

Die menschliche Kommunikation bedient sich indirekter Informationen aus den alltäglichen Situationen, um den Informationsfluss von Konversationen zu erhöhen. Diese indirekten Informationen werden auch als Kontext bezeichnet. Es gibt viele Definitionen dazu, was Kontext ist. Einige davon sind in Brown u. a. (1997), Schilit und Theimer (1994) oder Joshua u. a. (1998) zu finden. Für diese Arbeit wird die folgende Definition von Kontext und Kontextinformationen verwendet:

> *Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves.*
> *Anind K. Dey and Gregory D. Abowd*
> *Abowd u. a. (1999)*

Diese Definition besagt, dass jede für die Kommunikation relevante Information als Kontextinformation angesehen werden kann. Hieraus kann abgeleitet werden, dass nicht nur implizite, sondern auch explizite Informationen zur Kontextgewinnung herangezogen werden können. Um Kontextinformationen besser strukturieren zu können, wurden in Dey und Abowd im Rahmen des Projektes CyberDesk drei Kategorien definiert.

Physikalischer Kontext: In dieser Kategorie sind personenbezogene Objekte einzugliedern. Hierzu gehören die Position und Orientierung einer Person sowie deren Identität. Datumsinformationen wie Uhrzeit oder Tag gehören ebenfalls in diese Kategorie.

Handlungsorientierter Kontext: Beinhalten Informationen, welche die Aktionen einer Person beschreiben. Hierzu gehört unter anderem, was eine Person sagt oder denkt. Mit welcher Applikation oder Daten eine Person arbeitet gehört ebenfalls dazu.

Emotionaler Kontext: Hier wird der emotionale Zustand einer Person eingeordnet. Dies können beispielsweise die Zustände: glücklich, traurig, frustriert oder verängstigt sein.

Diese Kategorisierung ermöglicht die Einordnung personenbezogener Informationen, jedoch nicht Kategorisierung sämtlicher Informationen. Daher wird im Rahmen dieser Arbeit der physikalische Kontext um Sensorinformation der Umgebung erweitert. Dadurch können nicht nur personenbezogene, sondern auch umgebungsbezogene Daten, wie: Raumtemperatur,

Luftfeuchtigkeit, Wetterinformationen oder der Status von Haushaltsgeräten eingeordnet werden. Um auch verschiedene Informationen aus dem Internet zu kategorisieren, wird eine weitere Kategorie eingeführt.

Symbolischer Kontext: Hier werden Informationen eingeordnet, die sich aus dem Internet beziehen lassen. Beispielsweise sind dies Wetterinformationen sowie Wetterprognosen oder auch Statusänderungen aus sozialen Netzwerken.

3.2.2. Informationskategorisierung

In diesem Abschnitt der Arbeit sollen die relevanten Informationen der Szenarien extrahiert werden, um aus der Vielzahl der Informationen den Kontext abzuleiten. Dafür werden die Informationen in die definierten Kategorien aus Kapitel 3.2.1 eingeteilt.

Szenario: Familienurlaub

Aus diesem Szenario können die Informationen wie folgt kategorisiert werden:

Physikalischer Kontext

- *Position von Personen:*
  Aufgrund der Positionsinformationen können Rückschlüsse auf die Anwesenheit von Personen gezogen werden. Zusätzlich können ortsabhängige Dienste, wie die Öffnung der Tür, realisiert werden.
- *Pflanzenstatus:*
  Über die Feuchtigkeitssensoren an den Pflanzen können Rückschlüsse in Bezug auf die Pflege der Pflanzen geschlossen werden.

Handlungsorientierter Kontext

- *Bestimmung des Urlaubsstatus:*
  Der Status ist erforderlich zur Anpassung der Berechtigungsstruktur. Daten können beispielsweise aus dem Kalender oder der Position der Bewohner stammen.
- *Kürzliche Anmeldungen im System:*
  Die Anmeldung am System ist erforderlich, um die erlaubten Funktionen der Personen abzufragen und diese bereitzustellen. In diesem Beispiel wäre das für James die Funktion zur Öffnung der Tür.

Symbolischer Kontext

- *Bestimmung des Urlaubsstatus:*
  Informationen zum Status können ebenfalls aus sozialen Netzwerken stammen.

Szenario: Wie lange und welche Inhalte darf Carol im Fernsehen schauen

Die Daten aus diesem Szenario sind wie folgt eingeordnet:

Physikalischer Kontext

- *Position von Personen:*
  Aufgrund der Positionsinformationen können Rückschlüsse auf die Anwesenheit von Personen gezogen werden. Hierdurch kann bestimmt werden, vor welchem Gerät Carol TV schaut und ob sie in Gesellschaft ist.
- *Datum und Uhrzeit:*
  Aus diesen Informationen wird ebenfalls abgeleitet, ob der Fernseher genutzt werden darf.

Handlungsorientierter Kontext

- *Gerätestatus des TV:*
  Der Status ist erforderlich um weitere Informationen zu realisieren. Hierzu gehört vor allem, welche Sendungen Carol schaut und wie lange sie das Gerät in Betrieb hat.

Szenario: Oma June hat einen Kreislaufzusammenbruch

Die Informationen aus diesem Szenario wurden folgendermaßen gruppiert:

Physikalischer Kontext

- *Position von Personen:*
  Aufgrund der Positionsinformationen können Rückschlüsse auf die Anwesenheit von Personen gezogen werden. Des Weiteren können Positionsveränderungen registriert und Bewegungsprofile angelegt werden.

Handlungsorientierter Kontext

- *Geräteinteraktion:*
  Die explizite Interaktionen mit Geräten oder die Unterlassung von Aktionen kann Aufschluss über den Zustand der Person geben.

Szenario: Parkplatz - Sharing

In diesem Szenario sind die Daten wie folgt eingeordnet:

Physikalischer Kontext

- *Position von Personen:*
  Aufgrund der Positionsinformationen können Rückschlüsse auf die Anwesenheit von Personen gezogen werden.
- *Parkplatzbelegung*
  Aus dieser Information können Rückschlüsse gezogen werden, ob der Parkplatz vermietet werden kann.
- *Datum und Uhrzeit:*
  Aus diesen Informationen können Zeiträume für die Belegungsdauer abgeleitet werden.

Handlungsorientierter Kontext

- *Parkplatzfreigabe:*
  Die Freigabe des Parkplatzes für einen bestimmten Zeitraum erfolgt durch den Bewohner.

3.2.3. Informationsverarbeitung

Im vorherigen Abschnitt wurden die Informationen aus den Szenarien identifiziert und in Kategorien eingeordnet. Infolge dessen wird nun darauf eingegangen welche Handlungen aus den Informationen abgeleitet werden können und welche technischen Voraussetzungen beziehungsweise Konsequenzen daraus entstehen.

Positionsinformationen

Die Positionsdaten einer Person gehören zu den primären Kontextinformationen. Sie stellen ein Grundvoraussetzung für die vorgestellten Szenarien 3.1 dar, in dem sie die Realisierung der Szenarien erst ermöglichen oder ihre Sicherheit erhöhen. In den Szenarien 3.1.1, 3.1.3 und 3.1.4 kann die Position einer Person ein zusätzliches Sicherheitskriterium bei der Zutrittskontrolle darstellen. Da die Wohnungstür beziehungsweise das Garagentor, nur geöffnet werden kann, wenn die dazu berechtigte Person sich unmittelbar davor befindet. Des Weiteren können die Positionsinformationen genutzt werden, um Menschen in Gefahrensituationen zu unterstützen, wie in Szenario 3.1.3. In dem Szenario 3.1.2 können die Positionsinformationen unter anderem darüber Aufschluss geben, welche Geräte zurzeit verwendet werden und ob diese von einer

oder mehreren Personen genutzt werden.
Technisch gesehen gibt es eine Vielzahl von Möglichkeiten um die Position einer Personen zu erlangen und zu verfolgen. In Wohnräumen können Indoor-Positioning Systeme, wie beispielsweise das System der Firma Ubisense, verwendet werden. Anwendungsbeispiele für dieses System sind in (Voskuhl, 2011, S. 82 - 84) und Steggles und Gschwind zu finden. Für eine ungenauere und Personen unabhängige Lokalisierung wären auch Bewegungsmelder in den Räumen oder Sensoren in den Türschwellen denkbar. Die Ortung über das WLAN kann ebenfalls zur Ermittlung der Position verwendet werden. Außerhalb von Wohnräumen kann die Position von Personen über das in Smartphones integrierte GPS ermittelt werden.

Urlaubsstatus

Der Urlaubsstaus enthält die Information, ob die Familie im Urlaub ist.
Um dies realisieren zu können, muss die Information irgendwo hinterlegt sein. Eine Möglichkeit ist die Eintragung des Urlaubs in den Kalender, auf den das System Zugriff hat. Zusätzlich besitzt diese Möglichkeit den Vorteil, weitere Informationen in den Kalendereintrag abzulegen, wie beispielsweise Zutrittsberechtigungen. Eine weitere Art diesen Status zu identifizieren könnte zum einen die Analyse der Positionsdaten der Familienmitglieder sein, zum anderen können Informationen aus sozialen Netzwerken bezogen werden, wie beispielsweise durch Posts oder getaggte Fotos.

Datum und Uhrzeit

Bei diesen Informationen handelt es sich um sekundäre Kontextinformationen. Aufgrund dieser Daten ist es möglich die Nutzung von Geräten, wie beispielsweise den Fernseher in Szenario 3.1.2, zu kontrollieren. Ferner wird in Szenario 3.1.4 die Festlegung des Zeitraums, in dem der Parkplatz frei ist, realisierbar.
Technische Voraussetzung ist hierfür, dass die aktuellen Daten entweder im System vorhanden sind oder sie erfragt werden können, zum Beispiel aus dem Internet per NTP (Network Time Protocol, vgl. Mills u. a. (2010)).

Gerätestatus des TV

In Bezug auf Szenario 3.1.2 sind unter anderem Daten zur Ermittlung der Laufzeit und zu gezeigten Inhalten der TV-Geräte gemeint.
Um diese Daten aus einem TV-Gerät abfragen zu können, muss dieses über ein Netzwerkinterface verfügen, welches den Zugriff auf das Gerät ermöglicht. Eine Möglichkeit die

Informationen zu erfragen, beziehungsweise diese Geräte zu steuern, ist die Verwendung des UPnP- oder DLNA - Standards.

Pflanzenstatus

Diese Information bietet Unterstützung bei der Pflege von Pflanzen, indem der Anwender mitgeteilt bekommt, wann die Pflanzen Wasser benötigen. Des Weiteren können nützliche Informationen zur Pflanzenpflege, wie beispielsweise der Vorschlag eines Standortwechsels, gegeben werden.
Aus technischer Sicht sind hierfür zwei Sensoren notwendig. Zum einen ein Feuchtigkeitssensor, welcher es ermöglicht, die Feuchtigkeit des Bodens zu erfassen, zum anderen ein Fotowiderstand zur Protokollierung der Lichtintensität. Des Weiteren wird eine Netzwerkschnittstelle benötigt, um diese Daten auch propagieren zu können. In Verbindung mit einer Applikation, können die Sensordaten ausgewertet und dem Endanwender präsentiert werden.

Systemanmeldungen

Die Systemanmeldung ist eine wichtige Information. Sie gibt Auskunft darüber, ob eine ein Anwender angemeldet ist und wie lange schon.
Um eine Autorisation zu ermöglichen, wird eine Berechtigungsstruktur benötigt, welche die Nutzerdaten beinhaltet. Dies sind zum einen Daten, die eine eindeutige Identifikation einer Person ermöglichen, zum Beispiel Benutzername und Passwort. Zum anderen sind dies Daten über dessen Berechtigung. Die Ablage solcher Daten bietet beispielsweise das LDAP - Protokoll (Lightweight Directory Access Protocol), spezifiziert in K. Zeilenga und Foundation (2006) und J. Sermersheim und Novell (2006).

Parkplatzstatus

Die Information über den Parkplatz sagt aus, ob dieser besetzt oder frei ist. Aufgrund dieser Information kann dann entschieden werden, ob der Platz für eine Vermietung zur Verfügung steht.
Eine technische Möglichkeit zur Fahrzeugerkennung auf dem Parkplatz können Magnet- und Induktionssensoren sein.

3.3. Umgebung

Das hier zu entwickelnde Berechtigungssystem soll in das Living Place Hamburg integriert werden, welches schon in 2.1 vorgestellt wurde. Aus diesem Grund müssen die infrastrukturellen Gegebenheiten bei der Entwicklung berücksichtigt werden. Des Weiteren wird zusätzlich darauf geachtet, das System möglichst modular aufzubauen, um eine einfache Integration in andere Smart Homes zu ermöglichen.
Um spätere Designentscheidungen nachzuvollziehen, wird im Folgenden ein grober Überblick über die Sensorik, Aktorik und die Services des Living Place gegeben. Darauffolgend wird erläutert, wie diese Ressourcen miteinander kommunizieren.

3.3.1. Sensorik

Das Living Place Hamburg besitzt eine Vielzahl von Sensoren. Diese ermöglichen es, Informationen über die Umgebung zu sammeln und somit einen Kontext oder Kontextabhängigkeiten zu ermitteln. Des Weiteren werden die Sensoren für die Nutzbarkeitsanalysen verwenden. Die Sensoren werden im Folgenden kurz vorgestellt:

Positionserkennung Für die Positionserkennung wurde ein System der Firma Ubisense in das Living Place integriert. Über sogenannte Tags können Personen oder gegenständige innerhalb der Umgebung bis auf 15 cm genau lokalisiert werden, vgl. (Voskuhl, 2011, S. 82 - 84).

Couchsensorik Im Lounge-Bereich des Living Place befindet sich die Couch, welche mit kapazitiven Sensoren ausgestattet wurde. Hierdurch kann erkannt werden, wie viele Personen sich auf dem Sofa befinden und welche Position sie eingenommen haben. Hierdurch können beispielsweise Szenarien realisiert werden, die es ermöglichen, Lichtspots oder Lautsprecher automatisch auf Personen auszurichten, vgl. (Dreschke, 2011, S. 62 - 83).

Sturzerkennung Bei der Sturzerkennung handelt es sich um ein System welches durch die Kombination von kapazitiven Sensoren, die unter einem Teppich angebracht wurden und die Auswertung von Kamerabildern. Aus diesen Informationen kann anschließend auf einen Sturz und dessen Art geschlossen werden, vgl. Teske (2011) und (Dreschke, 2011, S. 40 - 51).

Intelligentes Bett Im Schlafbereich des Living Place wurde das Bett ebenfalls mit Sensoren bestückt. Dies ermöglicht eine berührungslose Detektion der Schlafphasen eines Menschen, vgl. Quast (2011) und Hardenack (2011). Zur Informationsgewinnung wurden

Dehnungsmessstreifen verwendet, wodurch keine elektromagnetischen Belastungen bei den Probanden verursacht werden.

Kameras Die im Living Place Hamburg eingebauten Kameras können vielseitig eingesetzt werden. Zum Beispiel für Gesichts- und Sturz- sowie Objekterkennung, vgl. Najem (2013). Ein weiteres Anwendungsgebiet für die Kameras stellen Usability-Untersuchungen dar. Insgesamt sind neun Kameras in die Umgebung integriert, welche teils statisch und teils schwenkbar sind. Die Daten können analog oder digital abgegriffen werden und laufen im Kontrollraum des Living Place zusammen.

Mikrofone Neben den Kameras sind ebenfalls fünf Richtmikrofone in der Nähe der Kameras angebracht worden. Somit ist es zusätzlich möglich Gespräche und Geräusche während einer Usability - Untersuchung zu dokumentieren oder eine Sprachsteuerung in das Living Place zu integrieren.

MESH Sensornetzwerk Das Sensornetzwerk besteht aus insgesamt acht Knoten. Sie dienen der Erfassung von Temperatur, Helligkeit und geben Auskunft über den Status der Fenster. Die stromsparenden Sensoren organisieren sich selber innerhalb eines MESH - Netzwerks und können um weitere Knoten erweitert werden, vgl. Pautz (2011).

3.3.2. Aktorik

Neben der Sensorik besitzt das Living Place Hamburg ebenfalls eine Vielzahl von Aktorik, die von normalen Anzeigegeräten bis hin zur Steuerung des Interieurs reicht. Die vorhandene Aktorik ist im Folgenden aufgeführt:

Lichtsteuerung In das Living Place wurden diverse RGB LEDs und Halogen Strahler eingebaut, um angenehme Lichtstimmungen zu schaffen und zu untersuchen, wie sich diese Lichtstimmungen und Licht allgemein auf die Probanden auswirkt. Dafür kann bei jedem Leuchtmittel die Farbe sowie die Intensität reguliert werden. Des Weiteren ist es möglich Szenen zu definieren, die beispielsweise einen Sonnenaufgang während der Aufwachphase simulieren oder die richtige Stimmung für ein romantisches Abendessen schaffen.

Fenstersteuerung Die Fenster wurden mit elektrischen Fensterantrieben ausgestattet. Somit kann jedes Fenster einzeln geschlossen und geöffnet werden. In Verbindung mit den Daten für Luftfeuchtigkeit und Raumtemperatur ist somit eine Regulierung des Raumklimas realisierbar.

Heizungssteuerung Durch die installierte Steuerung können die Heizkörper reguliert werden. Hierdurch ist eine individuelle und automatisierte Raumklimatisierung möglich. In Verbindung mit der Fenstersteuerung kann beispielsweise definiert werden, dass die Heizkörper beim Öffnen der Fenster abgeschaltet werden sollen.

Türsteuerung Ebenfalls wurde eine Steuerung der Wohnungstür in die Wohnumgebung integriert. Diese erlaubt das Öffnen der Wohnungstür per Applikation, vgl. (Bornemann, 2013, S. 2 - 4). Durch die Steuerung kann die Wohnungstür außerhalb der Wohnung per Smartphone oder per Applikation innerhalb der Wohnung geöffnet oder verschlossen werden.

Anzeigegeräte: Als reine Anzeigegeräte dienen die beiden Fernsehgeräte im Schlaf und im Lounge-Bereich der Wohnung. Auf ihnen werden Inhalte, wie das Fernsehprogramm oder das Videosignal der Türklingel, präsentiert.

3.3.3. Services

In der Informatik gibt es verschiedene Definitionen für den Begriff Service oder Dienst. Allgemein beschrieben bündelt ein Service eine oder mehrere Funktionen und stellt diese dem Anwender oder einer Applikation über Schnittstellen zur Verfügung. Dienste sollten unter anderem folgende Eigenschaften erfüllen: Technologieneutralität, Verwendung von Standards, lose Kopplung, Plattformunabhängigkeit und Ortsunabhängigkeit, vgl. Papazoglou (2003) und Papazoglou u. a. (2008).
Im Kontext dieser Arbeit werden unter dem Begriff Service alle Komponenten des Living Place eingeordnet, die aufgrund ihrer Funktionalitäten nicht eindeutig als Sensor oder Aktor identifiziert werden können.

Türklingel: Die im Living Place installierte Türklingel bietet den Besuchern der intelligenten Wohnumgebung die Funktionalitäten des Absetzens eines Klingelsignals, die Kommunikation mit dem Bewohner über Video oder das Hinterlassen einer Videomitteilung, vgl. Bornemann (2011). Dieses System ist auf der einen Seite ein Sensor (Abgabe des Klingelsignals) und auf der anderen Seite ein Aktor (Darstellung eines Video-Chats). Mit ihr können Services realisiert werden, wie zum Beispiel eine Zeit oder Personen unabhängige Klingelunterdrückung. Dies kann beispielsweise die Abschaltung der Klingelfunktionalität ab 22 Uhr sein.

Cubical: Bei dem Cubical handelt es sich um ein moderne Art der Fernbedienung, die eine nahtlose und einfache Bedienung von Heimgeräten ermöglichen soll. Entwickelt wurde

dieses Gerät, welches sich in der Kategorie der Tangible User Interfaces einordnet, von der Firma Seamless Interaction. Er wird zum einen als Sensor zur Steuerung von Geräten (über Kipp- und Drehbewegungen) genutzt. Zum anderen auch als Aktor über die Signalisierung eines Status über die Beleuchtung des Cubicals. In Verbindung mit der Türklingel kann beispielsweise der Service zur Bedienung der Türklingel sowie der Türsteuerung realisiert werden.

SiBar: Multitouch-Tresen und Multitouch-Couchtisch: Bei der SiBar handelt es sich ebenfalls um ein Projekt der Firma Seamless Interaction. Hier wurde eine Multitouch-Fläche von 200cm x 65cm in einen Küchentresen integriert. Auf ihm können Informationen, wie der Status der Wohnumgebung, Kalendereinträge, Nachrichten, etc. interaktiv dargestellt werden. Des Weiteren lassen sich die Aktoren des Living Place auch über den Tresen ansteuern. Zusätzlich bietet die SiBar den Service, die Inhalte aufgrund von Sensorinformationen zur Anwenderposition, neu anzuordnen.
Der Couchtisch wird ebenfalls zur Darstellung und Steuerung verwendet. Als Basis wurde hier der Microsoft Surface 2 verwendet. Wie der Couchtisch in das tägliche Leben eines Menschen integriert werden kann und wie Applikationen aufgebaut sein müssen um einfach und effektiv arbeiten zu können ist Thema aktueller und zukünftiger Arbeiten, vgl. Kühn (2012).

ISIS: Der *Indoor Spatial Information Service* zeichnet kontinuierlich Positionsdaten der Positionserkennung auf und sammelt somit Informationen über den Standort von Personen oder Gegenständen. In Verbindung mit einem 3D-Raummodell werden die Informationen semantisch angereichert. Durch diese semantischen Informationen können beispielsweise kontinuierliche Aufenthalts- und Bewegungsanalysen erstellt werden, vgl. Karstaedt (2012).

3.3.4. Kommunikationsschnittstelle

Bei der Entstehung des Living Place Hamburg wurde eine Kommunikationsstruktur gewählt, welche eine möglichst lose Kopplung zwischen den Ressourcen ermöglicht. Die Wahl fiel hier auf ein Message Broker System. Hierdurch können Ressourcen miteinander über einen zentralen Knotenpunkt kommunizieren, ohne sich gegenseitig zu kennen. Dies bedeutet: Der Sender weiß nichts über den Empfänger und umgekehrt, vgl. (Snyder u. a., 2011, S. 8). Im Living Place Hamburg wird der ActiveMQ von der Apache Software Foundation eingesetzt, da dieser über verschiedene Programmiersprachen angesprochen werden kann. Um Informationen zu erhalten oder zu publizieren, melden sich die vorher beschriebenen Ressourcen an verschiedenen

Topics bzw. Queues an, die für sie von Interesse sind. Der ActiveMQ hält diese Informationen vor und leitet eine eintreffende Nachricht an die entsprechenden Ressourcen weiter, vgl. Otto und Voskuhl (2010). Um verschiedene Kommunikationsarten realisieren zu können werden hier zwei unterschiedliche Mechanismen verwendet. Dies ist zum einen das Publish-Subscribe Pattern, zum anderen das Producer-Consumer Pattern. Das Publisher-Subscriber Pattern repräsentiert einen *One-to-Many* Mechanismus, bei dem der Sender seine Nachrichten über *Topics* verteilt. Hierzu melden sich Ressourcen an den für sie interessanten Topic an. Wird von Sender eine Nachricht an ein Topic geschickt, leitet der ActiveMQ diese Nachricht an die entsprechenden Ressourcen weiter. Dies ist in der Abbildung 3.1 nochmals dargestellt. Hierbei ist darauf zu achten, dass Nachrichten nicht zwischen gespeichert werden. Registriert sich eine Ressource kurz nach dem eintreffen einer Nachricht beim ActiveMQ, kann diese nicht mehr empfangen werden. Es kann erst die darauffolgend Nachricht weitergeleitet werden.

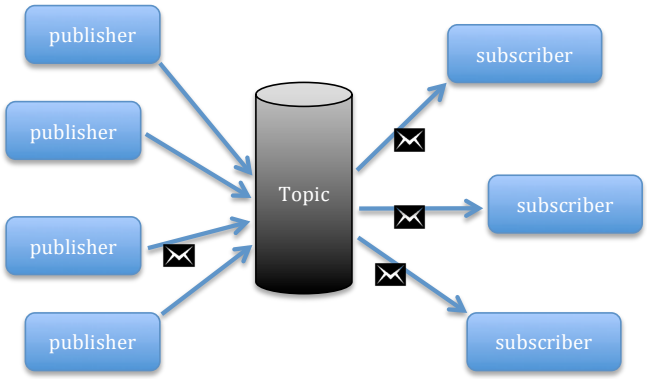

Abbildung 3.1.: Publish-Subscribe Pattern

Anders ist dies beim Producer-Consumer Pattern. Hier wird ein *Point-to-Point* Mechanismus abgebildet, bei dem der Sender seine Nachrichten über *Queues* versendet. Wird hier eine Nachricht an eine Queue versendet, wird diese Nachricht solange vorgehalten, bis sich eine Ressource an die entsprechende Queue des ActiveMQ anmeldet. So können keine Nachrichten verloren gehen. Sind mehrere Ressourcen an einer Queue angemeldet wird die empfangene Nachricht an denjenigen ausgeliefert, der sich zuerst registriert hat. Dies wird in der Abbildung 3.2 nochmals verdeutlicht. Weitere Informationen zu den eben vorgestellten Mechanismen sind in (Snyder u. a., 2011, S. 27, 28) zu finden.

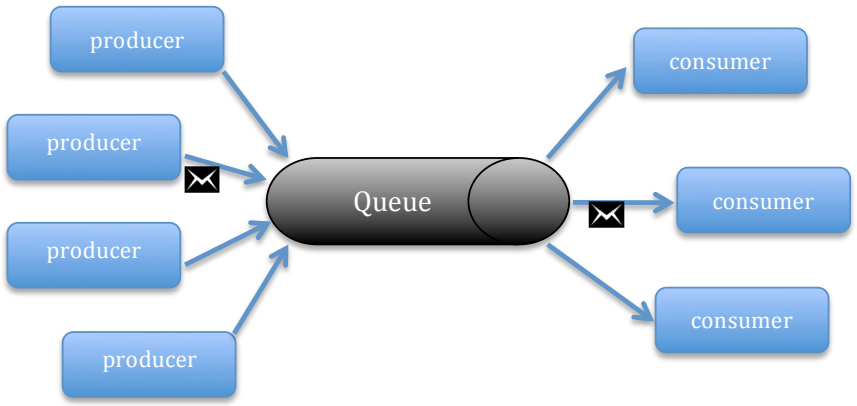

Abbildung 3.2.: Consumer-Producer Pattern

3.4. Verwandte Arbeiten

Mittlerweile sind auf der ganzen Welt Forschungslabore und Umgebungen geschaffen worden, welche im Bereich Ubiquitous Computing und Smart Home forschen. Hierzu gehören unter anderem The Georgia Tech Aware Home 3.4.1, MIT Intelligent Room und The Neural Network House at the University of Colorado Boulder, vgl. Mozer (1998). Aber auch in Deutschland gibt es einige Universitäten, die in diesem Bereichen forschen. Die bekanntesten sind in der SmartHome Initiative Deutschland zusammengefasst. Im Folgenden werden zwei verwandte Arbeiten vorgestellt und bewertet.

3.4.1. The Aware Home

Die Aware Home Initiative am Georgia Institute of Technology erforscht neuartige Sicherheitskonzepte für Applikationen in intelligenten Wohnumgebungen. Dafür wurde ein Haus gebaut und mit einer Vielzahl von Sensoren und Rechenkapazität ausgestattet. Diese werden genutzt, um Daten über die Bewohner zu gewinnen und anhand dieser Informationen über Aktivitäten abzuleiten. Da es sich hierbei auch um die Gewinnung sensibler Daten handelt, soll der Bewohner bestimmen können, wer Zugriff auf diese Daten erhält.
Um den Zugriff auf Daten zu kontrollieren, wird in diesem Projekt das RBAC-Modell verwendet. In diesem Modell werden zum einen Personendaten gespeichert, zum anderen auch Kontextinformationen in Form von Rollen abgelegt. Um dies zu erreichen wurde das RBAC -Modell um Umgebungsrollen (**Environment roles**) erweitert, vgl. Covington u. a. (2001). In diesen Rollen werden Umgebungsvariablen und Bedingungen festgehalten, die für einen

Zugriff auf eine Ressource erforderlich sind. Bei den Bedingungen kann es sich beispielsweise um Informationen wie eine Position, eine Uhrzeit oder einen bestimmten Tag handeln.
In der Abbildung 3.3 wird der Ablauf einer Zugriffsanfrage dargestellt. Wenn auf Daten oder

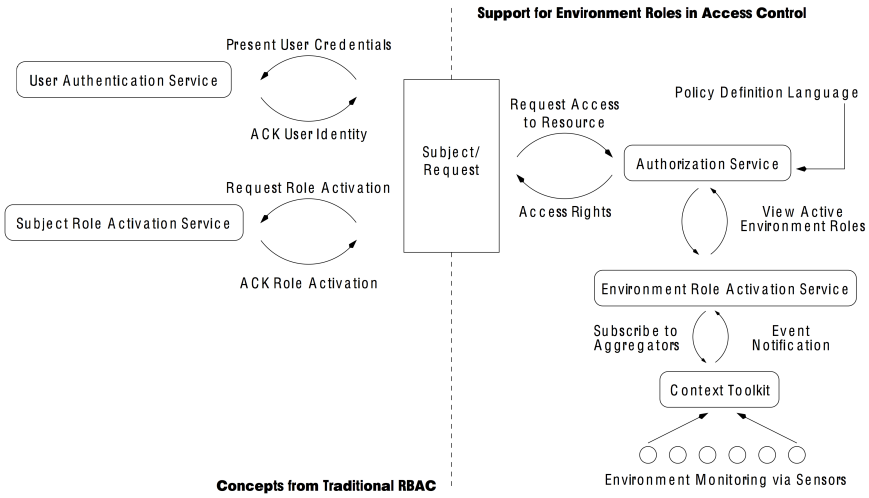

Abbildung 3.3.: Ablauf einer Zugriffsanfrage über Environment Roles

eine Ressource zugegriffen werden soll, wird diese Anfrage an den zentralen Autorisierungsservice weitergeleitet, welcher alle aktuellen Sicherheitsrichtlinien beinhaltet. Wenn für die aktuelle Anfrage eine Richtlinie existiert, wird der Environment Role Activation Service kontaktiert. Dieser prüft alle Bedingungen, die in der Environment Role angegeben sind und gibt diese zurück an den Autorisierungsservice. Stimmen diese Daten mit den Vorgaben aus der Sicherheitsrichtlinie überein, wird der Zugriff gewährt.

Bewertung

Die Aware Home Initiative verwendet das RBAC-Modell zur Zugriffsteuerung in intelligenten Wohnumgebungen und erweitert diese um eine Kontextsensitivität. Durch die Definition von Environment Roles können Daten in Rollen abgelegt werden. Dadurch ist das System sehr vielseitig und kann in andere Systeme integriert werden. Dies wird dadurch gewährleistet, dass die Erweiterungen in das traditionelle RBAC-Modell integriert wurden, ohne Veränderungen vorzunehmen. Des Weiteren können alle Eigenschaften und Vorteile des RBAC-Modells, wie

Hierarchisierung und Vererbung, genutzt werden.
Hauptaugenmerk des Frameworks ist die kontextsensitive Zugriffskontrolle auf sensible Daten. Hierbei wurde bisher jedoch keine Möglichkeit integriert, auch den Zugriff auf einzelne Funktionen einer Ressourcen zu reglementieren. Der Anwender hat entweder Zugriff oder eben nicht. Des Weiteren werden durch den *Environment Role Activation Service* nur jene Rollen aktiviert, die der aktuellen Situation entsprechen. Zeitliche Abhängigkeiten zwischen auftretenden Events, welche einen bestimmten Kontext beschreiben, werden nicht abgebildet.

3.4.2. MavHome

Das MavHome ist eine auf intelligenten Agenten basierende Umgebung. Diese hat es zum Ziel, den Komfort sowie die Effizienz der Bewohner zu steigern. Nachdem sich Anwender beim MavHome registriert haben, beginnt dies mit der Speicherung von Aktivitätsdaten zu der jeweiligen Person. Auf Basis dieser Daten werden unter anderem folgende Eigenschaften des MavHome realisiert: Vorhersagen über Personenverhalten, Erkennung von Personen aufgrund ihrer Aktivitäten, Mobilitätsvorhersagen, Anpassung der Multimediaumgebung und Unterstützung durch Robotik. Für die Kommunikation der Services untereinander wird CORBA eingesetzt.
Die Architektur des Agenten ist in vier Schichten unterteilt: die Entscheidungs-, Informations-, Kommunikations- und physische Schicht, vgl. Cook u. a. (2003). Der MavHome Agent kann zusätzlich in kleinere *lower-level*-Agenten zerlegt werden, welche sich um verschiedene Teilaufgaben kümmern können. Somit ist das Konzept auch für die Verwendung in größeren Umgebungen geeignet. Die folgende Abbildung 3.4 zeigt die Architektur des Agenten. Die Entscheidungsschicht wählt Aktionen für den Agenten aus, der diese anschließend ausführt. Welche Aktion gewählt wird, hängt von Informationen anderer Schichten, wie der Informationsschicht ab. Die Informationsschicht speichert und generiert Wissen, um daraufhin Entscheidungen zu treffen. Für den Austausch von Daten und Informationen zwischen den Agenten ist die Kommunikationsschicht zuständig. Die physikalische Schicht bildet die Schnittstelle zwischen den Agenten und der Hardware, welche sich in der Umgebung befindet, ab. Da die Architektur hierarchisch aufgebaut ist, kann hinter der physisch Schicht ebenfalls ein Agent verborgen sein.
Für die Gewinnung von Wissen werden in diesem Projekt hauptsächlich die Positionsdaten der Probanden verwendet. Jeder Proband besitzt ein Profil in dem aufgeführt ist, in welchen Räumen (Zonen) er sich aufgehalten hat und welche er anschließend aufgesucht hat. Hierdurch kann nachvollzogen werden welche Wege die Probanden am meisten gehen und warum. Durch die verwenden Lernalgorithmen können Bewegungen vorhergesagt werden. Detailtiere

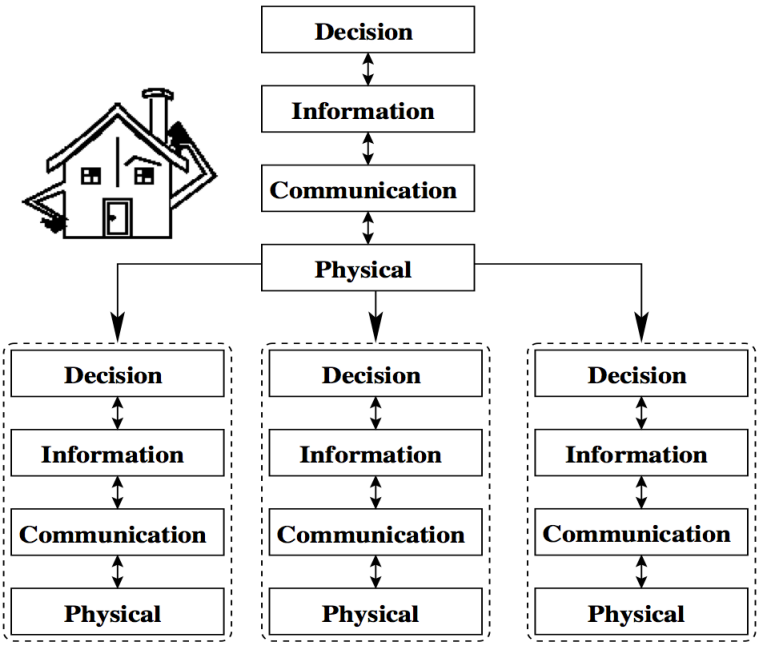

Abbildung 3.4.: MavHome - Agentenarchitektur

Informationen zu verwendeten Algorithmen und der Vorgehensweise sind in in den Artikeln von Roy u. a. (2007) und Das u. a. (2002) zu finden.

Bewertung

In diesem Projekt wird der Ansatz verfolgt, eine intelligente Wohnumgebung durch den Einsatz von Agenten zu schaffen. Auf Basis gesammelter Personendaten werden Vorhersagealgorithmen verwendet, um Aussagen über das zukünftige Verhalten der Anwender zu treffen. Die vorgestellte Architektur ermöglicht eine hohe Skalierbarkeit und kann somit in kleinen und großen Umgebungen eingesetzt werden.
Rückschlüsse auf eine aktuelle oder zukünftige Integration von Kontextinformationen konnte nicht festgestellt werden. Jedoch bietet die Architektur die Möglichkeit diese Informationen über die Informationsschicht aufzunehmen und in die Entscheidungsfindung einfließen zu lassen.

3.5. Abgrenzungen

Wie die vergleichbaren Arbeiten gezeigt haben, gibt es verschiedene Ansatzmöglichkeiten sich diesem Thema zu nähern. Des Weiteren wurde gezeigt, dass es hier nicht um ein triviales Themengebiet handelt. Aus diesem Grund muss definiert werden, welche Punkte diese Arbeit beinhaltet.
Das Hauptaugenmerk dieser Arbeit liegt auf der Entwicklung einer Berechtigungsstruktur für Smart Homes, welche den differenzierten Zugriff auf einzelne Funktionalitäten von Ressourcen ermöglichen soll. Darüber hinaus sollen Berechtigungen anhand von Kontextinformationen erteilt oder entzogen werden. Dies ermöglicht eine dynamische Anpassung von Berechtigungen in Abhängigkeit der aktuellen Situation.
Nicht zu dieser Arbeit gehört die Evaluierung diverser Systeme zur Kontextinterpretation und die Integration eines solchen Systems. Das Gleiche gilt für ein System zur Erfassung auftretender Events, welches beispielsweise durch regelbasierte Maschinen ermöglicht werden kann. Zudem wird das Anlegen sowie die Bearbeitung der Berechtigungsstruktur nur von Anwendern möglich sein, die zur Gruppe der Administratoren gehören. Hierfür wird eine Oberfläche zur Bearbeitung von Usern, Gruppen und Ressourcen bereitgestellt. Eine Unterstützung durch lernende Algorithmen aus dem Bereich der künstlichen Intelligenz sind im Moment nicht vorgesehen, können aber durch zukünftige Arbeiten hinzugefügt werden.

3.6. Fazit

Zum Anfang dieses Kapitels wurden Szenarien 3.1 definiert. Infolge dessen wurden auf Basis dieser Szenarien Anforderungen definiert, die zu einer Identifizierung des Kontextes führen. Hierfür wurde zunächst erläutert was genau Kontextinformationen sind und wie diese kategorisiert 3.2.1 werden können. Im Anschluss wurde die Umgebung vorgestellt, um einen Einblick darüber zu erlangen, welche Informationen das Living Place Hamburg zur Verfügung stellt und welche Funktionen und Services in Anspruch genommen werden können. Des Weiteren wurden zwei verwandte Arbeiten 3.4 vorgestellt und bewertet. Diese sollten mögliche Lösungsansätze für die Thematik dieser Arbeit verdeutlichen.

Ziel der Analyse war es, die Voraussetzungen und Anforderungen für ein kontextsensitives Berechtigungssystem zu identifizieren.
Die Voraussetzungen für solch ein System ergeben, dass Strukturen erforderlich sind, um die Daten, welche in einer Wohnumgebung anfallen, aufnehmen zu können. Zudem wird eine Komponente benötigt, die Informationen der Umgebung aufnehmen und diese miteinander

verknüpfen kann, damit eine Erkennung der aktuellen Situation, also des Kontextes, erfolgt. Darüber hinaus soll das System nicht nur die Zugriffskontrolle auf eine Ressource ermöglichen, sondern auch den Zugriff auf einzelne Funktionen dieser Ressource.
Die Erstellung und Bearbeitung von Berechtigungen soll durch den Anwohner erfolgen. Somit besitzt dieser die Kontrolle über das System.

4. Design

In diesem Kapitel wird ein Systementwurf vorgestellt, der die im Analysekapitel 3 identifizierten Anforderungen erfüllt. Hierbei wird das Ziel verfolgt, ein kontextsensitives Berechtigungssystem für Smart Homes zu erstellen.
Dafür wird zunächst ein Überblick 4.1 über das System gegeben. Darin wird erläutert, welche Voraussetzungen notwendig sind und welche Funktionen das System bereitstellen soll. Im Anschluss daran wird der Aufbau der Berechtigungsstruktur erläutert 4.2 und die damit einhergehende Organisation der Daten. Darauffolgend wird in Abschnitt 4.4 beschrieben, wie auf die im Living Place Hamburg vorhandenen Kontextinformationen reagiert werden kann. Des Weiteren wird in der Realisierung 4.5 auf die vorgestellten Konzepte eingegangen.

4.1. Überblick

Hinter dem hier zu entwickelndem Konzept steckt die Idee, einem Anwender beziehungsweise dem Bewohner eines Smart Homes, die Möglichkeit einzuräumen, den Zugriff auf digitale Services und Ressourcen zu kontrollieren. Dabei liegt der Fokus zum einen auf der Organisation der Daten, zum anderen auf einer dynamischen Zuweisung von Berechtigungen in bestimmten Situationen. Durch die Einbettung dieses Systems in das Living Place Hamburg und der Fähigkeit auf Kontextwechsel der Umgebung zu reagieren, unterscheidet sich dieses System von anderen Berechtigungssystemen.
Seit der Entstehung des Living Place Hamburg im Jahr 2009 sind diverse Projekte und Arbeiten entstanden. Einige dieser Arbeiten beschäftigten sich mit der Identifizierung von Kontextinformationen. Die Identifizierung von Kontextinformationen und daraus resultierende Kontexterkennung ist die Grundlage, auf der das Berechtigungssystem aufbaut. Aufgrund dieser Informationen können Kontextwechsel erkannt und entsprechend reagiert werden.
Um aus den Rohdaten, die von den im Living Place befindlichen Sensoren stammen, Kontextinformationen zu beziehen, sind verschiedene Schritte notwendig. Jene Arbeiten die sich mit dieser Thematik beschäftigt haben, werden im Folgenden kurz vorgestellt. Ein erster Schritt waren die Untersuchungen im Rahmen der Masterarbeit von Sören Voskuhl, vgl. Voskuhl (2011). In dieser Arbeit wurde untersucht, inwieweit der vorherrschende Kontext anhand

von modellunabhängigen Rohdaten erkannt werden kann. Mittels algorithmischer Verfahren konnten anhand von Rohdaten des Indoor-Postioning Systems Bewegungsprofile eines Bewohners erstellt werden. Mit der semantischen Anreicherung von Rohdaten anhand von Modellen beschäftigte sich Bastian Karstaedt, vgl. Karstaedt (2012). Durch die Verwendung von Gebäudemodellen konnten die Rohdaten des Indoor-Positioning Systems mit Informationen des Gebäudemodells verknüpft werden. Hierdurch wurde es möglich, nicht nur die Position einer Person als Information zu erhalten, sondern auch in welchem Raum oder vor welchem Gegenstand sich die Person befindet. Weiter führt die Arbeit von Jens Ellenberg, vgl. Ellenberg (2011). In dieser Arbeit werden die vorhandenen Informationen in einer Ontologie in Beziehung gesetzt. Somit können Aktivitäten in einem Smart Home erfasst und den Services zur Verfügung gestellt werden.

Durch diese Arbeiten ist die Voraussetzung, Kontext im Living Place Hamburg zu erkennen, geschaffen. Die Konsequenz daraus sollte sein, auf diese Informationen zu reagieren. Darin liegt der Ansatz des hier zu entwickelnden Berechtigungssystems. Das Berechtigungssystem erhält die Informationen aus dieser Deutungsebene und reagiert auf die vorherrschende Situation mit einer entsprechenden Anpassung der Berechtigungsstruktur. Um auf die aktuellen Situationen reagieren zu können, werden zunächst feste Regeln definiert. Diese geben vor, welche Kontextinformationen auftreten müssen, um eine Reaktion hervorzurufen. Auch das Anlegen der Daten in der Berechtigungsstruktur wird zunächst von Hand erfolgen. Für die Zukunft bieten sich jedoch Lernalgorithmen oder Verfahren aus dem Bereich der künstlichen Intelligenz an, um diese Arbeiten dem Anwender abzunehmen.

4.2. Berechtigungsstruktur

Die Berechtigungsstruktur dient der Ablage personenbezogener Daten. Dies reicht von Adressdaten, über Vorlieben bis hin zu Berechtigungen und Authentifizierungsdaten. Diese Daten müssen in einem System abgelegt und organisiert werden. Darüber hinaus ist ein geregelter Zugriff auf diese Daten für Personen, Systeme und Services in einem Smart Home erforderlich. Die Auswahl eines geeigneten Systems und die Organisation der Daten wird in den anschließenden Abschnitten dieses Kapitels diskutiert.

4.2.1. Verzeichnisdienst

Die Autorisation von Personen und deren Zuweisung von Berechtigungen stellt eine grundlegende Voraussetzung für dieses Projekt dar. Die Speicherung und Bereitstellung dieser Daten

wird meist über hierarchische Datenbanken realisiert. In Verbindung mit einem Netzwerkprotokoll entsteht somit ein Verzeichnisdienst, welcher die Daten zentral in einem Netzwerk zur Verfügung stellt. Bekannte Verzeichnisdienste sind beispielsweise das Active Directory von Microsoft, OpenLDAP oder Apache Directory, welche das LDAP-Protokoll zur Speicherung und Abfrage von Daten nutzen.
In dieser Arbeit kommt das Apache Directory (ApacheDS) zum Einsatz, da dieses einige Vorteile mit sich bringt. Ein Vorteil ist zum Beispiel, dass der Server in Java geschrieben ist und somit in beliebigen Java-Komponenten und als eigenständiger Server betrieben werden kann. Des Weiteren bietet das Projekt Installationspakete für Windows, Mac OSX und diverse Linux-Distributionen an. Dies bietet eine hohe Flexibilität und Unabhängigkeit bei der Wahl der Umgebung. Zudem unterstützt der ApacheDS Kerberos 5 und das Change Password Protocol. In der aktuellen Version 2.0 stehen weitere Eigenschaften wie: Dynamische Schemata, LDIF basierte Konfiguration, SASL / StartTLS und eine bessere Performance zur Verfügung, vgl. The Apache Software Foundation.

4.2.1.1. Informationsmodell

Die Informationen werden in einer hierarchischen Datenbank abgelegt. Somit ergibt sich eine Baumstruktur bei der Ablage der Daten. An der obersten Stelle steht der Directory Information Tree (DIT). Unterhalb des DIT werden die Knoten definiert, welche im Folgenden als Einträge bezeichnet werden. Die Einträge beherbergen wiederum Attribute, welche die tatsächlichen Informationen in Form von Schlüssel-Werte-Paaren enthalten. Welche Attribute in einem Eintrag abgelegt werden können oder müssen, wird in sogenannten Schemata festgelegt. Auf die Schemata wird im folgenden Abschnitt 4.2.1.2 näher eingegangen.
Die Einträge unterhalb des DIT können in zwei Kategorien eingeteilt werden. Zum einen sind dies Einträge die weitere Untereinträge als Attribute beinhalten. Diese dienen zur Strukturierung der Informationen, indem die Informationen in bestimmte Bereiche oder in Anlehnung an die Unternehmensstruktur gegliedert werden. So können beispielsweise die Informationen unterschiedlicher Standorte eines Unternehmens in Domänen zusammengefasst werden. Die Abteilungen innerhalb des Standortes können in Organisationseinheiten strukturiert werden. Zum anderen existieren Einträge, welche keine Untereinträge beherbergen. Diese Speichern in ihren Attributen Informationen zu Objekten der realen oder virtuellen Welt, wie beispielsweise Daten von Personen, vgl. Zörner (2008).

Da im Living Place solche Strukturen nicht existieren, muss eine Aufteilung anhand anderer Kriterien erfolgen. Aufgrund der in der Analyse definierten Anforderungen ergibt sich

die in Abbildung 4.1 dargestellte Baumstruktur. Alle Informationen werden unter der Do-

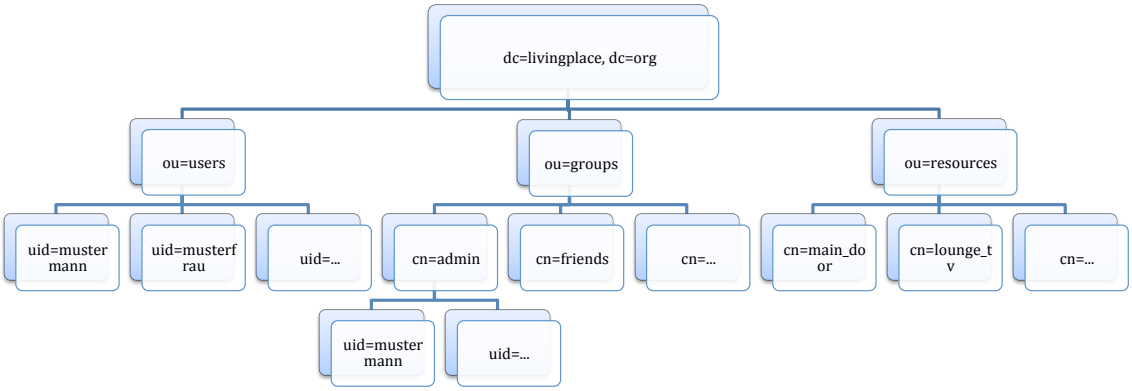

Abbildung 4.1.: LDAP Baumstruktur

mäne *dc=livingplace, dc=org* zusammengefasst. Unterhalb der Domäne werden die Daten in Organisationseinheiten unterteilt. Hierdurch entstehen die Einheiten User, Gruppen und Ressourcen.

User

In dieser Organisationseinheit werden personenabhängige Informationen abgelegt. Welche Schlüssel-Werte-Paare zu einer Person gespeichert werden können, wird durch Schemata festgelegt. In Abbildung 4.2 ist ein beispielhafter Eintrag dargestellt. Die oberen fünf Attribute

Attributbeschreibung	Wert
objectclass	***inetOrgPerson (strukturell)***
objectclass	***LPUsers (strukturell)***
objectclass	***organizationalPerson (strukturell)***
objectclass	***person (strukturell)***
objectclass	***top (abstrakt)***
cn	**Sven Boris Bornemann**
sn	**Bornemann**
description	Student an der HAW
jpegPhoto	JPEG-Bild (239x264 Pixel, 44579 Bytes)
LPCurrentPlace	lounge
LPDoorBellID	351554051112705894902000007655062 38
uid	bornemann@de.de
userPassword	SSHA gehashtes Passwort

Abbildung 4.2.: Beispieleintrag eines Users

vom Typ *objectclass* beschreiben, welchen Schemata der User unterliegt. Darauffolgend sind

die aktuellen Attribute aufgeführt. Zu diesen Daten gehören auch die Zugangsdaten einer Person, welche für die Autorisation verwendet werden. Der Benutzername ist unter der Attributbezeichnung *uid* und das Passwort unter *userPassword* gespeichert. Hierbei ist zu beachten, dass durch die hier dargestellten Schemata weitere Daten, wie beispielsweise die Adressdaten oder Telefonnummern, abgelegt werden können. Durch die Erweiterung oder das Hinzufügen von Schemata ist die Ablage weiterer Attribute möglich.

Gruppen

Die Organisationseinheit der Gruppen dient der Verknüpfung von Personen und Ressourcen. Dafür werden in dem Eintrag ein oder mehrere Personen anhand ihrer *uid* aufgeführt. Somit ist die Verbindung zu den Personen hergestellt. Des Weiteren werden jene Berechtigungen definiert, die diese Personen ausüben dürfen. In Abbildung 4.3 ist ein Beispieleintrag in diesem Format zu sehen. Wie auch bei den Personeneinträgen sind hier als Erstes die verwandten

Attributbeschreibung	Wert
objectclass	***groupOfUniqueNames (strukturell)***
objectclass	***LPGroups (strukturell)***
objectclass	***top (abstrakt)***
cn	**admin**
uniqueMember	**uid=bornemann@de.de**
uniqueMember	**uid=peter@mustermann.de**
uniqueMember	**uid=petra@musterfrau.de**
LPRight	{"edit_user":["all"],"main_door":["open","close"],"lounge_tv":["on","off"],"light_control":["all"]}

Abbildung 4.3.: Beispieleintrag einer Gruppe

Schemata aufgeführt. Darauffolgend wird der Name der Gruppe, die zugehörigen Personen und deren Berechtigungen aufgelistet.
Die Ablage der Berechtigungen liegt in einem speziellen Format vor. Es handelt sich hierbei um eine im JSON-Format serialisierte HashMap. Diese Designentscheidung wurde aufgrund der vorherrschenden Kommunikationsstruktur des Living Place Hamburg getroffen. Der geringe Overhead bei der Kommunikation mit dem JSON-Format begünstigt diese Entscheidung zusätzlich. Durch dieses Format können die Berechtigungsinformationen ohne weitere Bearbeitung über eine *Queue* oder einen *Topic* des ActiveMQ versandt werden. Darüber hinaus bietet diese Art der Datenablage noch weitere Vorteile. Zum einen können hierdurch mehrere Berechtigungen verschiedener Ressourcen in einem Attribut abgelegt werden. Dies verringert die Komplexität der LDAP-Abfragen. Zum anderen wird die Bearbeitung der Berechtigungen vereinfacht, da diese nach einer Deserialisierung wieder als HashMap zur Verfügung stehen und somit leicht zu verarbeiten sind.

Ressourcen

Die in der Analyse 3 identifizierten Aktoren, Sensoren und Services werden ebenfalls in einer Organisationseinheit zusammengefasst. Es handelt sich hierbei um die Ressourcen. Wie die Personen haben die verschiedenen Ressourcen ebenfalls wichtige Informationen, welche in der Berechtigungsstruktur abgelegt werden. Neben einem eindeutigen Namen gehört auch die Auflistung der ansprechbaren Funktionen, der Typ der Ressource und ihre Kommunikationsschnittstellen dazu. Mit Kommunikationsschnittstelle ist in diesem Fall der Queue- oder Topic-Name gemeint, über den die Ressource ansprechbar ist. Diese Daten können beispielsweise für eine direkte Kommunikation mit einer Ressource verwendet werden, ohne dabei den ActiveMQ zu nutzen. Über die Information, welche Funktionen die Ressource bereitstellt, kann beim Anlegen einer Berechtigung verifiziert werden, ob diese Berechtigung auch zulässig ist. Infolge der gesammelten Daten kann ebenfalls ein Service Discovery Dienst bereitgestellt werden, der anfragenden Applikationen Informationen zu den verschiedenen Services bereitstellen kann.
In Abbildung 4.4 ist ein Beispieleintrag der Ressource *main_door* dargestellt.

Attributbeschreibung	Wert
objectClass	***LPResources (strukturell)***
objectClass	***top (abstrakt)***
LPResName	**main_door**
jpegPhoto	JPEG-Bild (797x796 Pixel, 28374 Bytes)
LPResFunctions	["open","close"]
LPResIP	172.16.0.55
LPResQueue	doorcontrol
LPResType	actor

Abbildung 4.4.: Beispieleintrag einer Ressource

4.2.1.2. Schemata

Jeder Verzeichnisdienst bringt eine Anzahl unterschiedlicher Schemata mit. Sie dienen der Beschreibung der Einträge, da in ihnen die Form und Struktur festgelegt ist. Somit kann der Verzeichnisdienst bei der Erstellung oder Bearbeitung die Gültigkeit eines Eintrags anhand der Schemata verifizieren. Bei der Erstellung von Schemata werden folgende Bestandteile definiert:

- Attribute
- Objektklassen
- Syntaxregeln

- Vergleichsregeln

Jedes Schema ist eindeutig anhand des *Object Identifier* (OIDs) identifizierbar. Die OIDs bestehen aus einer unbeschränkten Zeichenkette aus Zahlen, die mit einem Punkt getrennt sind. Sie sind ebenfalls hierarchisch gegliedert, wodurch neue OIDs durch anhängen einer weiteren Zahl generiert werden. OIDs sind weltweit eindeutig und werden durch die Internet Assigned Numbers Authority (IANA) verwaltet. Organisationen beantragen hier auch neue OIDs. Im Falle der HAW Hamburg lautet die Basis-ID *1.3.6.1.4.1.21841.1*, welche für diese Arbeit durch anhängen weiterer Zahlen erweitert wurde, vgl. Zörner (2008).
In der Analyse 3 wurden bestimmte Informationen identifiziert, welche das Erstellen eigener Schemata erforderlich machte. Die definierten Attribute und Objektklassen werden im Folgenden vorgestellt.

Attribute

Es wurden Attribute angelegt, welche Informationen und Zustände über Personen und Ressourcen aufnehmen können. Eine Beschreibung dieser ist im Folgenden zu finden.

LPDoorBellID - *OID: 1.3.6.1.4.1.21841.1.2.1 - deprecated*
: Dieses Attribut wurde am Anfang als Authentifizierungsmerkmal verwendet. Mit Einführung von Gruppen und Ressourcen wurde das Attribut allerdings überflüssig.

LPDoorBellElligibility - *OID: 1.3.6.1.4.1.21841.1.2.2 - deprecated*
: Dieses Attribut wurde ebenfalls in einem sehr frühen Stadium der Arbeit definiert. Dort wurden Berechtigungen zum Öffnen beziehungsweise Schließen der Wohnungstür definiert. Dies wird aktuell durch die Zuweisung von Gruppen realisiert.

LPResType - *OID: 1.3.6.1.4.1.21841.1.2.3*
: Hier wird der Typ einer Ressource abgelegt. Dies können entweder Sensoren, Aktoren oder Services sein.

LPResName - *OID: 1.3.6.1.4.1.21841.1.2.4*
: In diesem Attribut wird der Name der Ressource festgehalten, welcher zur Identifizierung verwendet wird.

LPResFunctions - *OID: 1.3.6.1.4.1.21841.1.2.5*
: Hier werden alle Funktionen einer Ressource aufgeführt, die von anderen Ressourcen verwendet werden können.

LPResQueue - *OID: 1.3.6.1.4.1.21841.1.2.6*
Enthält den Namen der Queue, über die jene Ressource erreichbar ist.

LPResTopic - *OID: 1.3.6.1.4.1.21841.1.2.7*
Enthält den Namen des Topic, über die jene Ressource erreichbar ist.

LPAllergy - *OID: 1.3.6.1.4.1.21841.1.2.8*
Dieses Attribut wurde definiert, um Informationen zu Allergien einer Person zu speichern. Zurzeit wird dieses Attribut nicht verwendet, kann aber von zukünftigen Services, wie einem intelligenten Kochbuch, verwendet werden.

LPCurrentPlace - *OID: 1.3.6.1.4.1.21841.1.2.9*
In diesem Attribut wird die aktuelle Position einer Person innerhalb der Wohnung gespeichert.

LPRight - *OID: 1.3.6.1.4.1.21841.1.2.10*
Dieses Attribute dient der Ablage von Berechtigungen und wird in den Gruppen verwendet

LPResIP - *OID: 1.3.6.1.4.1.21841.1.2.11*
Hier wird die momentane IP-Adresse der Ressource gespeichert. Diese kann abgefragt werden, falls mit der Ressource kommuniziert werden soll, ohne dabei die Mechanismen des ActiveMQ zu verwenden.

LPOnline - *OID: 1.3.6.1.4.1.21841.1.2.12 - deprecated*
Dieses Attribut wurde für die Verwaltung der Berechtigungen definiert. Um dort anzuzeigen, welcher Anwender zurzeit Online ist und das System verwendet.

Objektklassen

Um die definierten Attribute zu organisieren wurden Objektklassen erstellt. Diese ermöglichen die Ablage von Informationen, gemäß dem erstellten Informationsmodell. Im Folgenden werden die Objektklassen erläutert.

LPUsers - *OID:1.3.6.1.4.1.21841.1.1.1*

Die Objektklasse LPUsers dient der Speicherung personenabhängiger Daten. Dazu werden Eigenschaften des Objektes *inetOrgPerson* geerbt, da dieses Objekt Felder zur Speicherung des Namen, Benutzernamen, Passwort, E-Mail, Adresse, Telefonnummern, etc. zur Verfügung stellt. Des Weiteren sind eigens definierte Attribute im Objekt definiert, welche die Verwaltung zusätzlicher Daten einer Person ermöglicht. Dies sind Daten zur aktuellen Position einer Person innerhalb der Wohnumgebung und bekannte Allergien. Darüber hinaus können in Zukunft weitere Attribute definiert werden, welche die Ablage von Hobbys, Musikgeschmack und weitere Vorlieben einer Person ermöglichen. Die Abbildung 4.5 zeigt einen Ausschnitt des eben vorgestellten *LPUsers*-Objekts.

```
objectclass ( 1.3.6.1.4.1.21841.1.1.1
    NAME 'LPUsers'
    DESC 'User im LivingPlace'
    SUP inetOrgPerson
    STRUCTURAL
    MUST ( uid $ userPassword )
    MAY ( LPAllergy $ LPCurrentPlace )
)
```

Abbildung 4.5.: LDAP - Objekt: LPUsers

LPGroups - *OID:1.3.6.1.4.1.21841.1.1.2*

Dieses Objekt stellt die Verknüpfung zwischen den Anwendern und den Ressourcen der Umgebung her. Es wird geregelt, welche Benutzer welche Berechtigungen besitzen. Dazu werden die Eigenschaften von *groupOfUniqueNames* geerbt. Somit können Attribute vom Type *uniqueMember* hinzugefügt werden, welche die User der Gruppe repräsentieren. Des Weiteren wird durch dieses Attribute gewährleistet, dass ein Anwender nur einmal in der Gruppe auftreten darf. Dies verhindert eventuelle Doppelbelegungen und dadurch entstehende Inkonsistenzen. Darüber hinaus existiert das Attribut *LPRight*, in dem die Berechtigungen der Gruppe festgehalten sind. Der Aufbau des *LPGroups*-Objekts ist in der Abbildung 4.6 zu sehen.

```
objectclass ( 1.3.6.1.4.1.21841.1.1.2
    NAME 'LPGroups'
    SUP groupOfUniqueNames
    STRUCTURAL
    MUST uniqueMember
    MAY LPRight
)
```

Abbildung 4.6.: LDAP-Objekt: LPGroups

LPResources - *OID:1.3.6.1.4.1.21841.1.1.3*

Das *LPResources*-Objekt beinhaltet ausschließlich Attribute zur Speicherung von Geräteinformationen. Hierzu zählen unter ande-

```
objectclass ( 1.3.6.1.4.1.21841.1.1.3
    NAME 'LPResources'
    STRUCTURAL
    MUST LPResName
    MAY ( LPResFunctions $ LPResIP $ LPResQueue $ LPResTopic $ LPResType $ jpegPhoto )
)
```

Abbildung 4.7.: LDAP-Objekt: LPResources

rem die Gerätefunktionen, die IP-Adresse, der Standort, der Gerätetyp, etc.
Ob weitere Attribute in das Objekt aufgenommen werden müssen, ist zum aktuellen Zeitpunkt nicht abschätzbar. Der bisherige Aufbau des Objekts ist in der Abbildung 4.7 zu sehen.

4.3. Complex Event Processing

Die Datenstruktur zur Erfassung und Speicherung der Daten ist durch die vorgestellte Berechtigungsstruktur 4.2 definiert. Um auf einen Kontextwechsel reagieren zu können, wird ein reaktives Verhalten innerhalb des Berechtigungssystems benötigt. Dies kann auf diverse Arten realisiert werden. Eine Möglichkeit besteht in der Verwendung einer regelbasierten Maschine. Diese besitzen jedoch meist keine Möglichkeit zeitliche Abhängigkeiten aufzulösen. Für die Kontexterkennung in diesem Umfeld sind zeitliche Korrelationen allerdings unerlässlich. Daher ist der Einsatz eines Complex Event Processing Systems erforderlich.
Unter Complex Event Processing (CEP) versteht man in der Informatik die Erkennung und Verarbeitung von Ereignissen, während diese gerade auftreten. Hierbei liegt der Fokus auf der Identifizierung komplexer Zusammenhänge aus unterschiedlichen Datenströmen. Darüber hinaus können CEP-Systeme zeitliche und kausale Abhängigkeiten zwischen Ereignissen erkennen. Die Identifizierung komplexer Events aus einzelnen Ereignissen und den Beziehungen zwischen diesen Ereignissen ist alles andere als trivial. Um komplexe Events zu erkennen werden zum Beispiel Mechanismen wie endliche Automaten und gefärbte Petri Netze verwendet, vgl. Walzer u. a. (2007).
Gerade für den Einsatz in Smart Home Umgebungen sind CEP-Systeme interessant, da die Erkennung einer bestimmten Situation von mehreren Vorfällen abhängen kann, die wiederum in einer bestimmten zeitlichen Abfolge auftreten müssen. Die Verwendung von CEP-Systemen ist nicht nur in Smart Home Umgebungen interessant, sondern beispielsweise auch im Echtzeit-Aktienhandel oder bei der Überwachung komplexer Systeme.
Im Allgemeinen beschreiben Ereignisse oder auch Events Aktivitäten in einem System und beinhalten drei Aspekte, vgl. Luckham (2002).

Gestalt Events treten in Gestalt eines Objekts auf. Dabei können Events in simpler Form, beispielsweise als String oder in komplexer Form mit mehreren Datentypen auftreten. In komplexeren Events kann beispielsweise enthalten sein, zu welchem Zeitpunkt diese auftreten, ob sie ein periodisches Verhalten zeigen oder ob sie mit anderen Ereignissen in Beziehung stehen.

Signifikanz Ein Event beschreibt die Signifikanz einer Aktivität. Rückschlüsse darauf können aus der Gestalt des Events gezogen werden, da die enthaltenen Daten meist Aufschluss über die Signifikanz geben.

Relativität Aktivitäten stehen meist über die Zeit, Kausalität oder Aggregationen in Zusammenhang. Events besitzen die gleichen Beziehungen wie Aktivitäten und können somit über dieselben Eigenschaften in Beziehung gesetzt werden. Beziehungen zwischen Events werden als Relativität bezeichnet.

4.3.1. Drools Fusion

Drools Fusion ist ein Projekt der JBoos Community. Durch dieses Modul wird es ermöglicht, ein CEP-System in eine Java Applikation zu integrieren. Drools Fusion ermöglicht die Erfassung von Events aus unterschiedlichen Datenströmen, wie beispielsweise Dateien, JMS Queues oder Sockets.
Drools Fusion besitzt zwei Modi zur Verarbeitung von Events. Der Modi namens *Cloud Mode* ist der Standard Verarbeitungsmodus, in dem alle Objekte der aktuellen Faktenbasis als Fakt interpretiert werden. Dabei spielt es keine Rolle, ob es sich um eine Event oder einen tatsächlichen Fakt handelt. Eine Kenntnis über zeitliche Abhängigkeiten der Fakten ist in diesem Modus nicht vorhanden. Im Gegensatz dazu steht der *Stream Mode*. Dieser Modus wird für die Verarbeitung von Datenströmen verwendet. Die Daten werden dafür mittels eines Zeitstempels zeitlich geordnet. Hierdurch können Events aus unterschiedlichen Datenströmen synchronisiert und in zeitliche Abhängigkeit gebracht werden.
Drools Fusion wurde im Rahmen der Masterarbeit von Kjell Otto in das Living Place Hamburg integriert. Die Wahl viel auf dieses Projekt, da sich die gute Dokumentation und die offenen Quellen des Open Sources Projektes sehr gut für den Einsatz in einer Laborumgebung eignen. Des Weiteren ist die Programmiersprache mit der Drools Fusion verwendet werden kann identisch mit der im Living Place genutzten Sprache Java. Hiermit wird der Aufwand für die Nutzung minimiert, vgl. Otto (2013).

4.3.2. Events

In Drools Fusion sind Ereignisse, auch Events genannt, ein spezieller Typ von Fakten. Dies bedeutet alle Events sind Fakten, jedoch sind nicht alle Fakten Events. Die Charakterisierung von Events wird durch folgende Punkte beschrieben, vgl. (The JBoss Drools team, 2012, S. 5f.):

Usually immutables: Dies bedeutet, Events sind unveränderlich. Die Unveränderlichkeit begründet sich auf der Definition von Events. Diese beschrieben eine Zustandsänderung

in einer bestimmten Anwendungsdomäne. Sobald ein Event aufgetreten und registriert wurde, liegt es bereits in der Vergangenheit und ist somit unveränderlich.

Strong temporal constraints: Dieser Punkt beschreibt die zeitliche Abhängigkeit von Events in Regeln. In einer Regel können nur zeitliche Korrelationen zwischen Events hergestellt werden.

Managed lifecycle: Aufgrund der Unveränderlichkeit und zeitlichen Korrelation von Events ist die Verbindung von Ereignissen nur in einem begrenzten Zeitraum möglich. Dies ermöglicht der CEP-Maschine eine automatische Verwaltung der Lebenszyklen von Ereignissen. Die Dauer in der ein Event nach dessen auftreten noch gültig ist, wird durch die *@expieres*-Direktive innerhalb der Regel definiert. Nach Ablauf der definierten Zeit ist das Event hinfällig und wird nicht mehr verwendet.

Use of sliding windows: Auftretende Events werden mit einem Zeitstempel versehen. Dadurch können über einen gewissen Zeitraum, beispielsweise 60 Minuten, gleitende Zeitfenster für Bedingungen und Aggregationen definiert werden. Dies ermöglicht zum Beispiel die Ermittlung einer durchschnittlichen Raumtemperatur oder die durchschnittliche Anzahl von Personen in einem Raum.

4.4. Dynamische Berechtigungsanpassung

In den vorherigen Abschnitten wurde auf die verwendeten Systeme eingegangen. Dies war zum einen die definierte Berechtigungsstruktur 4.2. Diese statische Struktur dient der Speicherung von Daten. Zum anderen wird ein Complex Event Processing System 4.3 eingesetzt, welches die dynamische Komponente des Systems widerspiegelt. Hierdurch kann auf aktuelle Situationen reagiert werden.
Aufgrund der Kombination dieser beiden Systeme ergeben sich zwei unterschiedliche Ansätze zur Reaktion auf einen Kontextwechsel. Zum einen kann die Berechtigung direkt in dem Verzeichnisdienst ein- oder ausgetragen werden. Zum anderen kann die resultierende Konsequenz einer Berechtigung über die Regeln im CEP ausgeübt werden, ohne dabei die Berechtigungen im Verzeichnisdienst anzupassen. Die beiden Lösungsansätze werden in den folgenden Abschnitten erläutert und bewertet.

4.4.1. Indirekte Steuerung

Bei dieser Art der Ereignisreaktion wird die Berechtigungsstruktur direkt angepasst. Dies bedeutet: Wird ein Kontext aufgrund der vorliegenden Informationen erkannt, wird die hierfür

definierte Berechtigung zu einer Person oder Ressource hinzugefügt oder entfernt.
Anhand des Szenarios Familienurlaub aus Abschnitt 3.1.1 soll der Ablauf näher erläutert werden. In diesem Szenario wird James, abhängig vom aktuellen Kontext, die Berechtigung zum Öffnen der Wohnungstür erteilt oder entzogen. Dies ist abhängig von zwei Informationen. Zum einen muss die Familie im Urlaub sein, zum anderen muss James direkt vor der Tür stehen. Treffen diese beiden Kontextinformationen zu, wird James zur Gruppe *main_door_control* hinzugefügt, wodurch ihm die Berechtigung zugewiesen wird. Er kann die Funktionen nun über die Smart Home App verwenden. Verlässt James die Wohnung, treffen die Kontextinformationen nicht mehr zu und James wird aus der Gruppe wieder entfernt. Das Sequenzdiagramm 4.8 stellt den geschilderten Ablauf noch mal dar.
Dieses Vorgehen bietet unter anderem den Vorteil, dass die Berechtigung nur zugewiesen

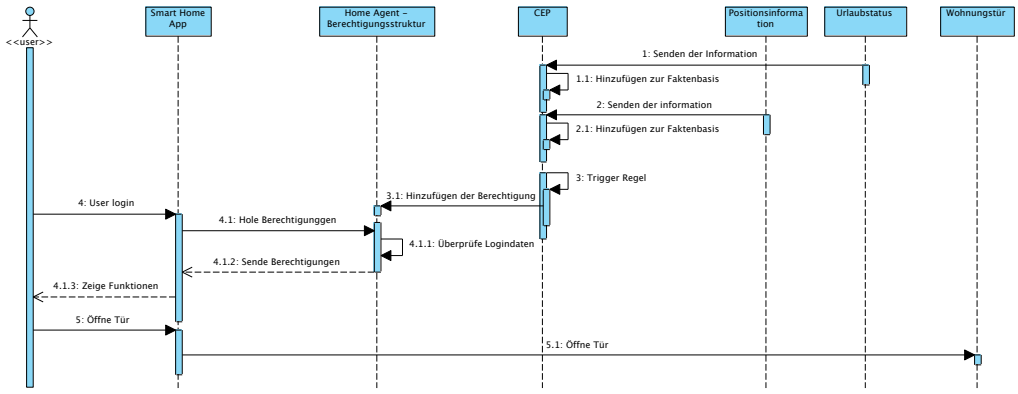

Abbildung 4.8.: Sequenzdiagramm: Beispiel zur indirekten Steuerung

werden, wenn der Kontext eintritt. Dies erhöht die Sicherheit des Systems. Allerdings wird hierdurch auch die Komplexität gesteigert, da jede Zuweisung einer Berechtigungen durch eine Regel im CEP definiert werden muss. Somit sind bei diesem Ansatz die Regeln personengebunden. Ausgehend vom Beispiel der Wohnungstürsteuerung, müssen also pro Person zwei Regeln definiert werden: Eine zum Eintragen und eine zum Löschen der Berechtigung.

4.4.2. Direkte Steuerung

Im Gegensatz zur vorher beschriebenen Art der Ereignisreaktion bleibt hier die Berechtigungsstruktur unangetastet. Tritt eine Kontextänderung auf, wird die betroffene Ressource direkt angesteuert.
Am Beispiel des Szenarios Familienurlaub bedeutet dies Folgendes: James möchte über die Smart Home App die Wohnungstür öffnen. Der Befehl wird vom CEP-System registriert und

in die Faktenbasis übernommen. Stimmen die Kontextinformationen mit den Vorgaben in der Regel überein, sendet das System den Befehl zum Öffnen der Tür an die Türsteuerung. Die Kommunikation zwischen den Komponenten wird im Sequenzdiagramm 4.9 dargestellt.
Aufgrund dieses Vorgehens, wird nur eine Regel im CEP-System benötigt, da diese die Aufgabe

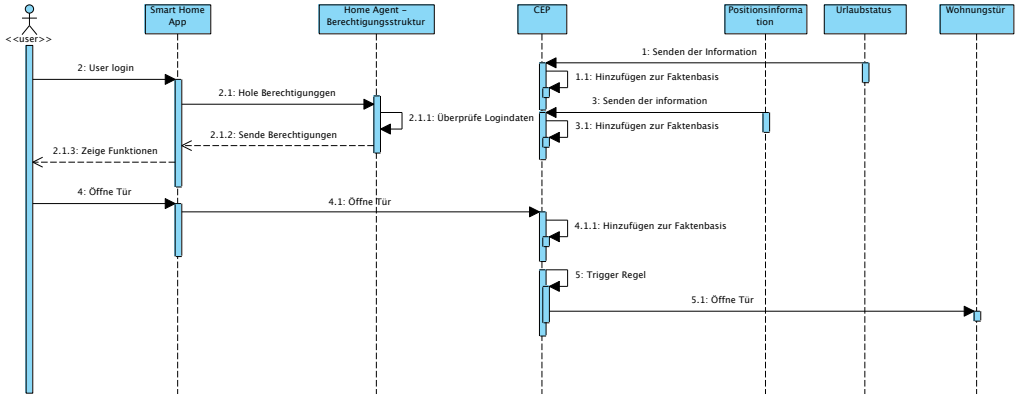

Abbildung 4.9.: Sequenzdiagramm: Beispiel zur direkten Steuerung

hat, eine Ressource anzusteuern. Die Regel ist nun also nicht Personen-, sondern Ressourcenabhängig. Dies verringert die Anzahl der Regeln und die Komplexität des Systems. Nachteil dieses Verfahrens ist, dass die Berechtigung vorher in die Berechtigungssteuerung eingetragen werden muss. Dadurch kann der Anwender die Funktionen auch auswählen, obwohl der Kontext zurzeit nicht besteht. Jedoch wird die Ausführung durch die Regel im CEP-System verhindert.
Im Hinblick auf die spätere Benutzung durch den Endanwender und der genannten Vorteile, wurde dieser Ansatz gewählt. Um diesen Ansatz zu verdeutlichen ist im nachfolgenden Abschnitt ein Beispiel aufgeführt.

4.4.3. Beispiel: Öffnen der Wohnungstür

In den Szenarien 3.1 des Analysekapitels wurden diverse Ereignisse identifiziert, welche für die Erkennung der aktuellen Situation und der anschließenden Reaktion erforderlich sind. Um den Vorgang der direkten Steuerung zu verdeutlichen, soll dieser anhand des Beispiels zum Öffnen der Wohnungstür erläutert werden. Die benötigten Informationen hierfür müssen über den ActiveMQ bezogen und anschließend der Faktenbasis hinzugefügt werden. Um dies zu ermöglichen, muss ein Programmcode, im Folgenden Sensor genannt, implementiert werden, welcher sich an den entsprechenden Topic beziehungsweise Queue des ActiveMQ anmeldet. Somit können die aktuellen Informationen abonniert werden. Des Weiteren muss ein Pro-

grammcode, im Folgenden Aktor genannt, implementiert werden, der die Ansteuerung der elektronischen Türsteuerung aus den Regeln heraus ermöglicht.
Eine genauere Beschreibung der implementierten Sensoren und Aktoren folgt in der Beschreibung des Living Place Controlling 4.5.2.2.

Informationsgewinnung

Für das Beispiel sind zwei Informationen von Relevanz. Zum einen die Information, ob eine Person gerade die Tür öffnen möchte. Zum anderen kann die Positionsinformation dieser Person herangezogen werden. Diese wird nicht unbedingt benötigt, kann sich jedoch positiv auf die Sicherheit des Systems auswirken, da durch diese Information die Wohnungstür nur geöffnet wird, wenn sich die Person direkt davor befindet. Somit wird eine eventuell ungewollte Öffnung verhindert.
Beide Informationen können im Living Place über den ActiveMQ bezogen werden. Daraus resultiert, dass zwei Sensoren implementiert werden müssen, die diese Informationen bei ihrem Auftreten erfassen und zur Faktenbasis hinzufügen. Im Anschluss erfolgt die Interpretation der erhaltenen Daten durch die im CEP-System vorhandenen Regeln.

Reaktion

Nachdem die Sensoren zur Erfassung der Daten vorhanden sind, wird noch ein Aktor benötigt, der das CEP-System mit der elektrischen Türsteuerung verbindet. Dieser Aktor wird ausgeführt, sobald die für dieses Beispiel definierte Regel zutrifft. Ist das der Fall, sendet der Aktor den entsprechenden Befehl zur Öffnung der Tür über den ActiveMQ zur elektronischen Türsteuerung, welche den Befehl dann ausführt.
Die Verbindung zwischen Sensoren und Aktoren wird über eine Regel hergestellt. Durch die Verknüpfung beider Sensorinformationen wird der aktuelle Kontext erkannt. Mit dem darauffolgendem Ausführen des Aktors wird das gewünschte reaktive Verhalten erzeugt. Die dafür benötigte Regel ist in Abbildung 4.1 dargestellt.

```
1 import org.livingplace.infrastructure.homeagent.utils.
     SpatialObject;
2 import org.livingplace.bundles.messagingdefines.
3             messages.Messages;
4
5 global  org.livingplace.infrastructure.homeagent.
6                          actions.ControlDoor ControlDoorAction;
7
```

```
8  declare Messages
9      @role ( event )
10     @expires( 30s )
11 end
12 declare SpatialObject
13     @role ( event )
14     @expires( 30s )
15 end
16
17 rule "Control Door Rule"
18     dialect "mvel"
19     when
20         m : Messages(operation == "open")
21         p : SpatialObject((ObjectType == "Door") &&
22              (InSpace == true) &&
23              (this before[ 0s, 30s ] m))
24
25     then
26         System.out.println("fire door control rule");
27         ControlDoorAction.controlDoor(m);
28
29 end
```

Listing 4.1: Regel zur Öffnung der Wohnungstür

Um die von den Sensoren erfassten Objekte in der Regel verwenden zu können, müssen diese zunächst importiert werden. Anschließend wird der Aktor bekannt gemacht, um im späteren Verlauf die gewünschten Funktionen aufrufen zu können. Da das System im Stream Modus verwendet wird, werden Eigenschaften für die verwendeten Datentypen definiert. Dabei wird definiert, dass es sich um Events handelt. Dies bewirkt den Aufruf der Regel, sobald eines der Objekte von einem Sensor erfasst und der Faktenbasis hinzugefügt wurde. Des Weiteren wird eine Gültigkeitsdauer für die Events verwendet. Darauffolgend beginnt die Regel mit der Vergabe eines Namens und dem verwendeten Dialekt. Im anschließenden Bedingungsteil von Zeile 19 bis 24 werden die Bedingungen für die Identifizierung des Kontextes festgelegt. Wie zuvor erwähnt, ist das der Befehl zum Öffnen der Tür und die aktuelle Position der Person. Des Weiteren wird eine zeitliche Abhängigkeit zwischen diesen Informationen definiert. Der Kontext ist nur erfüllt, wenn sich die Person vor dem Absenden des Öffnungsbefehls in der Nähe der Tür aufhält und das Auftreten dieses Events nicht älter als 30 Sekunden ist. Anders gesagt, wenn sich der Bewohner vor der Tür befindet, hat er 30 Sekunden Zeit, diese zu öffnen.

Ansonsten bleibt die Tür verschlossen. Ist diese Bedingung erfüllt, wird der Befehl aus Zeile 27 ausgeführt und die Tür öffnet sich. Die Erläuterung zu möglichen Funktionen einer Regeln und der verwendeten Syntax ist im *Drools Fusion User Guide* nachzulesen, vgl. The JBoss Drools team (2012).

4.5. Realisierung

Basierend auf den im vorherigen Abschnitt vorgestellten Strukturen, wird nun die konkrete Umsetzung des Systems erläutert. Diesbezüglich wird zunächst ein Überblick über das Gesamtsystem 4.5.1 geschaffen, in dem die Komponenten kurz vorgestellt und ihre Interaktion untereinander erläutert werden. Anschließend werden die Teilkomponenten 4.5.2 im Detail vorgestellt.

4.5.1. Systemüberblick

Das Gesamtsystem besteht aus diversen Einzelkomponenten, welche miteinander agieren, um die in den Szenarien 3.1 identifizierten Funktionen bereitstellen zu können. Abbildung 4.10 stellt alle hierfür relevanten Komponenten dar. Die Schnittelle zwischen dem System und dem Endanwender bilden die beiden Komponenten *Smart Home App* und *User Management* ab. Die Smart Home App 4.5.2.4 wurde im Zuge zweier Projektarbeiten entwickelt und diente anfänglich ausschließlich zu Steuerung eines Türschlossantriebs, vgl. Bornemann (2013) und Bornemann (2012). Im Laufe dieser Arbeit wurde diese Applikation jedoch um die Funktionalitäten zur Anzeige von Berechtigungen erweitert. Die darauffolgende Inanspruchnahme der freigegebenen Ressourcen wurde für die Ansteuerung der Türschlosssteuerung realisiert. Der Ausbau von Funktionen ist vorerst nicht angedacht, da die Funktionalitäten dieser Applikation mit dem System des User Management 4.5.2.3 zusammengeführt werden sollen. Hinter dem User Management steht eine Web-Applikation, welche die Verwaltung von Usern, Gruppen und Ressourcen des Berechtigungssystems ermöglicht. Vorteil dieser Applikation ist ihre Plattformunabhängigkeit, da sie über eine Browser Applikation von unterschiedlichen Betriebssystemen und Geräten aufgerufen werden kann. Aus diesem Grund ist auch die Übernahme der Funktionalitäten aus der Smart Home Applikation geplant.
Der Home Agent 4.5.2.1 dient als Schnittstelle zum Verzeichnisdienst. Über diese Schnittstelle können Informationen über Ressourcen sowie Sensoren und Aktoren, aber auch Daten zu Personen abgerufen und bearbeitet werden.
Das Living Place Controlling 4.5.2.2 ermöglicht die Erkennung eines Kontextes und das anschließende Auslösen einer Reaktion.

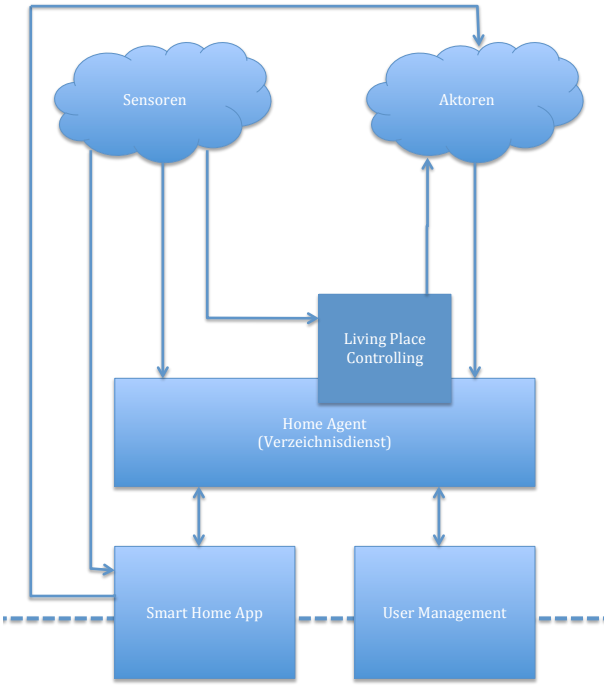

Abbildung 4.10.: Gesamtübersicht

Die Sensor- und Aktorwolke symbolisiert die Vielzahl verschiedener Sensoren und Aktoren. Eine Auflistung und Erläuterung, der im Living Place befindlichen Sensoren und Aktoren wurde in 3.3 vorgenommen.

Am Anfang der Realisierung wurde das Ziel verfolgt, alle Komponenten des Systems als OSGi-Bundle zu implementieren. Dies hätte verschiedene Vorteile.
Zum einen hätten die bereits vorhandenen Mechanismen des Living Place Controlling-Bundles, zu dem auch das CEP-System gehört, genutzt werden können.
Zum anderen hätte das System mit seinen verteilten Komponenten von einer Stelle aus überwacht und gesteuert werden können. Dies scheiterte jedoch an der Implementierung des ActiveMQ Client-Bundles, welches den OSGi-Bundles die Kommunikation mit dem ActiveMQ ermöglichen sollte. Ein Grund dafür war der Aufbau der Abhängigkeiten der ActiveMQ Client Bibliotheken. Jene konnten von den Mechanismen des verwendeten OSGi-Frameworks nicht aufgelöst werden, da diese nicht im richtigen Format vorlagen. Um dieses Problem zu lösen, wäre ein händischer Nachbau von Nöten gewesen. Dies stellte sich aber als zu großer Aufwand heraus, da dieser Vorgang bei jeder neuen Version der ActiveMQ Client Bibliothek wieder-

holt werden müsste. Aufgrund dessen wurden die einzelnen Komponenten vorerst nicht als OSGi-Bundles implementiert. Jedoch sind die Komponenten so konzipiert, dass sie zu einem späteren Zeitpunkt mit geringem Aufwand in OSGi-Bundles umgewandelt werden können. Infolgedessen mussten Teile des Living Place Controlling Bundles extrahiert und in den Home Agent integriert werden, um die Funktionalitäten des CEP-Systems zu nutzen. Ersichtlich wird dies auch im Klassendiagramm des Home Agent 4.12, welches zu einem späteren Zeitpunkt erläutert wird.

4.5.1.1. Kommunikationsmechanismen

Die Kommunikation zwischen den in Abbildung 4.10 dargestellten Komponenten wird über das Message Broker System ActiveMQ realisiert. Zwischen diesen Komponenten erfolgt die Informationsübermittlung ausschließlich über Queues. Dadurch kann sichergestellt werden, dass die Nachrichten nicht verloren gehen, da der ActiveMQ die Nachrichten in einer Queue solange aufbewahrt, bis sie der Empfänger abholt, im Gegensatz zu einem Topic. Somit wird die Verarbeitung der Nachrichten garantiert. Die Funktionsweisen von Queues und Topics wurden in 3.3.4 veranschaulicht.

Nachrichtenformat

Für die Kommunikation mit dem Verzeichnisdienst über den ActiveMQ wurde zunächst ein Nachrichtenformat definiert. In diesem werden zum einen alle derzeit möglichen Funktionen abgebildet. Zum anderen soll das Format zukünftige Funktionen ebenfalls abbilden können oder diesbezüglich erweiterbar sein.
Um die Handhabbarkeit und die Programmierung zu vereinfachen, wurden zunächst Java Objekte angelegt, welche die erforderlichen Daten aufnehmen können. Die Informationen können in den Objekten abgelegt und mit Hilfe der Gson-Bibliothek von Google zu einem JSON-String serialisiert und deserialisiert werden. Somit können die Objekte mit allen erforderlichen Daten über den ActiveMQ versendet werden.
Wie in dem Klassendiagramm 4.11 dargestellt ist, lehnen sich die Objekte an die verwendete LDAP Baumstruktur 4.1 des Verzeichnisdienstes an. Die Gruppen aus dem Verzeichnisdienst werden durch die Klasse *LPGroup* repräsentiert. Für die Ressourcen und Personen steht ebenfalls ein passendes Pendant zur Verfügung. In diesen drei Klassen werden die jeweiligen Informationen für oder aus dem Verzeichnisdienst abgelegt. Diese Klassen werden in der *Messages* Klasse, gebündelt. Des Weiteren sind in dieser Klasse zusätzliche Attribute vorhanden, welche für die Kommunikation von Relevanz sind.
Das Attribut *operation* beinhaltet den auszuführenden Befehl. Das kann zum Beispiel das

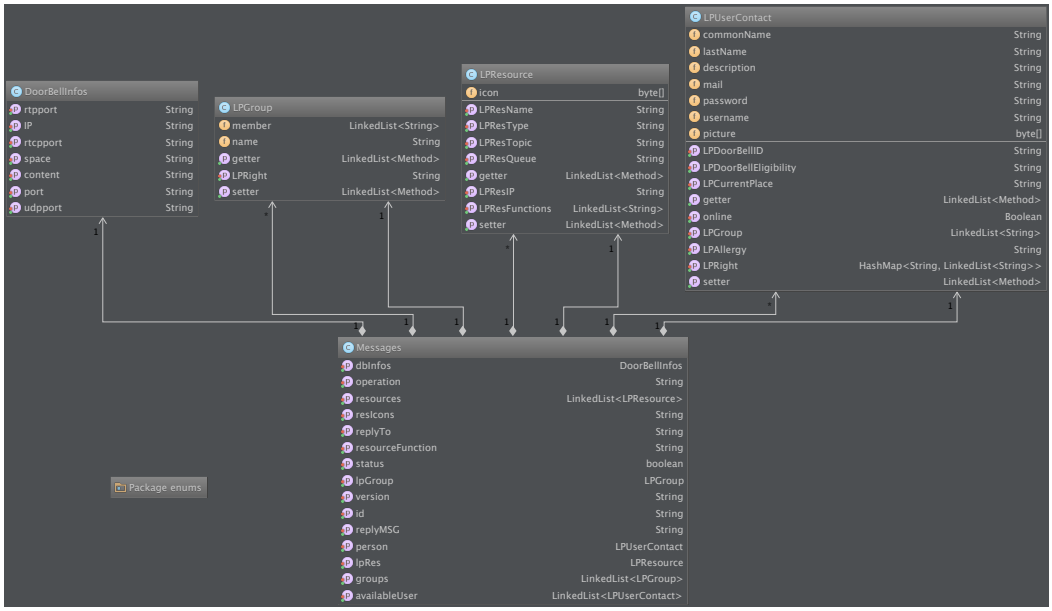

Abbildung 4.11.: UML Klassendiagramm des Bundles Messagingdefines

Anlegen eines Users oder das Entfernen einer Berechtigung in einer Gruppe sein. Weitere, für die Kommunikation wichtige, Attribute sind *status* und *replyTo*. Im ersten Attribut wird über ein Boolean-Wert signalisiert, ob der zuvor gesendete Befehl korrekt ausgeführt wurde. Über das zweite Attribut wird dem Kommunikationspartner mitgeteilt über welches Topic beziehungsweise welche Queue eine Antwort gesendet werden soll.

4.5.2. Teilkomponenten

Nachdem ein grober Überblick über das Gesamtsystem geschaffen wurde, werden in diesem Abschnitt die einzelnen Teilkomponenten diskutiert. Um ein besseres Verständnis der Komponenten zu erlangen, wird zum einen der Aufbau und die Funktionalitäten der Komponenten beschrieben. Zum anderen wird erläutert, wie die Komponenten untereinander kommunizieren.

4.5.2.1. Kommunikationsschnittstelle des Home Agent

In den vorherigen Abschnitten des Kapitels wurde der Aufbau des gewählten Verzeichnisdienstes vorgestellt. Da der Verzeichnisdienst in die Kommunikationsstruktur des Living Place integriert werden soll, wird im Folgenden der Home Agent vorgestellt. Dieser fungiert als Schnittstelle zwischen dem ActiveMQ, also allen Aktoren, Sensoren und Services des Living

Place und dem Verzeichnisdienst.
In Abbildung 4.12 ist das Klassendiagramm des Home Agent abgebildet. In der Initialisierungsphase werden zunächst alle notwendigen Verbindungsparameter aus der *settings.xml* abgerufen, zu denen unter anderem die IP-Adressen vom ActiveMQ und dem Verzeichnisdienst gehören. Da es sich hier um ein verteiltes System handelt, kann der Home Agent auf einem beliebigen Rechner im Netzwerk gestartet werden. Alle benötigen Parameter dafür sind in der settings.xml vorhanden. Anschließend wird mit der Klasse *CConsumer* ein Thread gestartet, der sich an der Queue *personbase* anmeldet und darüber die Nachrichten zur Bearbeitung des Verzeichnisdienstes abonniert.
Sobald eine Mitteilung eintrifft, wird diese an die *decodeJSON*-Methode aus der Klasse *JSONUtils* weitergereicht. Dort wird die auszuführende Operation aus der Nachricht extrahiert und die entsprechende Methode zum Bearbeiten der Operation aufgerufen. Das Ergebnis der Methodenaufrufe wird über die in der Mitteilung enthaltene Antwort-Queue zurück an den Absender geschickt. Hierfür wird die Klasse *CProducer* verwendet. Für die Kommunikation und Datenhaltung wird das in 4.5.1.1 vorgestellte Nachrichtenformat verwendet.
Die Verbindung zum Verzeichnisdienst wird über die Methoden der Klasse *LDAPConnections* hergestellt. Um Daten von Personen zu manipulieren, werden die Methoden der Klasse *LDAPContactMethods* verwendet. Analog dazu, fungieren die Klassen *LDAPGroupMethods* und *LDAPResourceMethods* zur Bearbeitung von Gruppen und Ressourcen.
Für die Verbindung zum Verzeichnisdienst wird die LDAP -Bibliothek des Spring Frameworks verwendet, vgl. SpringSource. Die LDAP Spring Bibliothek kapselt die traditionelle LDAP-Kommunikation und lenkt somit den Fokus auf die Bearbeitung der Daten.

Der gewählte Ansatz bietet unter anderem den Vorteil, dass der Verzeichnisdienst genutzt werden kann, ohne dass sich der Programmierer in die Kommunikationsmechanismen dieses Dienstes einarbeiten muss. Das jeweilige Versenden von Nachrichten im JSON-Format über den ActiveMQ reicht diesbezüglich aus. Ein weiterer Vorteil ist, dass die Operationen, welche auf dem Verzeichnisdienst ausgeführt werden, begrenzt werden können. Da die Schnittstelle nur diejenigen Operationen bereitstellt, die von Services, Sensoren, Aktoren und anderen Programmen ausgeführt werden dürfen.

4.5.2.2. Living Place Controlling

Mit dem Verzeichnisdienst wurde eine Struktur geschaffen, die das Speichern von Informationen und Berechtigungen ermöglicht. Ein reaktionäres Verhalten auf Basis vorhandener

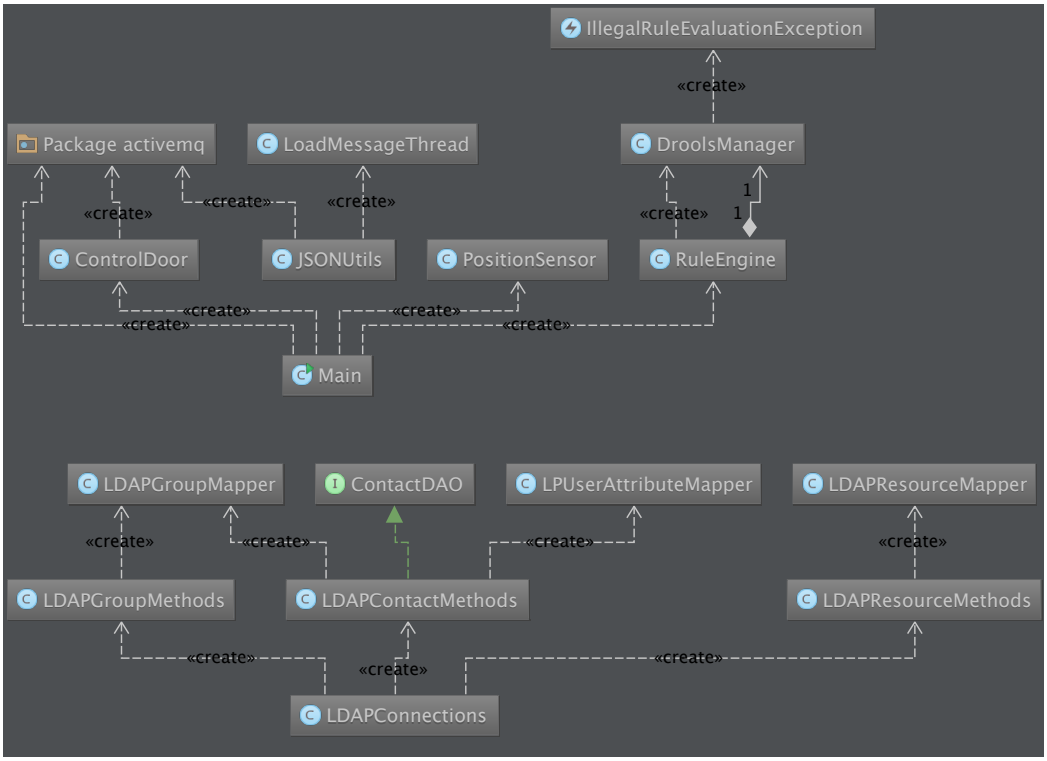

Abbildung 4.12.: UML Klassendiagramm des Home Agent

Sensorinformationen kann hierdurch aber nicht realisiert werden. Um auf einen Kontext reagieren zu können, wird daher ein Complex Event Processing System verwendet.
Das CEP-System wurde im Rahmen der Masterarbeit von Kjell Otto (vgl. Otto (2013)) in das Living Place integriert. Das sogenannte Living Place Controlling beherbergt das vom JBoss Team entwickelte CEP-System Drools Fusion, in dessen Faktenbasis die Informationen abgelegt werden können. Detaillierte Erläuterungen zu Drools Fusion wurden in 4.3 vorgenommen. Das Living Place Controlling läuft in einer OSGi Umgebung und ist selbst in verschiedene OSGi-Bundles unterteilt. Details zur Implementierung und zum Aufbau des Living Place Controlling sind im Realisierungsabschnitt der Masterarbeit von Kjell Otto zu finden, vgl. (Otto, 2013, S. 56-71).
Wie schon in 4.5.1 erläutert, konnte diese Struktur nicht verwendet werden, da es Probleme mit der Erstellung eines ActiveMQ-Client-Bundles gab. Stattdessen wurden Komponenten aus dem Living Place Controlling extrahiert und in den vorher beschriebenen Home Agent 4.5.2.1 integriert. Ersichtlich wird dies auch durch die Klassen *RuleEngine* und *DroolsManager* aus dem gezeigten Klassendiagramm 4.12 des Home Agent.
Um die Realisierung der Szenarien zu ermöglichen, wurde das Living Place Controlling um zwei Sensoren und einem Aktor erweitert. Im Folgenden werden jene vorgestellt, die für die Realisierung des Szenarios Familienurlaub 3.1.1 benötigt werden. Aus zeitlichen sowie technischen Gründen wird der Urlaubsstaus aus diesem Szenario nicht über einen implementierten Sensor ermittelt, sondern als gegeben angesehen. Des Weiteren beschränkt sich die Realisierung auf dieses Szenario. Weitere Szenarien aus dem Kapitel 3.1 wurden nicht implementiert, können jedoch durch das realisierte System zu einem späteren Zeitpunkt leicht hinzugefügt werden.

PositioningSensor: Dieser Sensor sammelt Informationen über die Position von Personen im Living Place und schreibt diese Informationen in die Faktenbasis. Um die Informationen zu bekommen, abonniert das Bundle ein bestimmtes Topic des ActiveMQ, auf dem die Positionsdaten zur Verfügung gestellt werden.
Die Position einer Person ist eine grundlegende Kontextinformation für alle vorgestellten Szenarien 3.1.

ControlDoor: In dieser Klasse wurden Sensor und Aktor vereint. Der Sensor erfasst Informationen, die über die Queue *doorControl* gesendet werden und schreibt diese in die Faktenbasis des CEP-Systems. Der Aktor dient der Ansteuerung des Türschlosses. Sobald die Regel zutrifft, dass jemand die Tür öffnen möchte und die Person auch vor der Tür steht, wird aus der Regel heraus der Aktor ausgeführt. Im Aktor wird der Steue-

rungsbefehl über die entsprechende Queue des ActiveMQ an den Türschlossantrieb gesendet.

4.5.2.3. User Management

In den beiden vorherigen Abschnitten wurden Komponenten des Systems vorgestellt, welche für die Realisierung und Bereitstellung der Funktionen verantwortlich sind. In diesem Abschnitt geht es um jene Komponente, die es dem Endanwender ermöglicht, sein System zu konfigurieren beziehungsweise die Berechtigungen zu verwalten.

Bei dieser Komponente handelt es sich um eine Web-Applikation, die mittels des Web-Frameworks Play erstellt wurde. Das Play-Framework ermöglicht eine einfache Erstellung komplexer und dynamischer Web-Applikationen mit Java und Scala. Einige Vorteile des Frameworks bestehen in der Nutzung altbewährter Java Bibliotheken und der Auflösung von Abhängigkeiten durch Maven. Wie auch andere Web-Frameworks verwendet Play das MVC-Pattern. MVC-Pattern steht für Model View Controller. Die Idee hinter diesem Entwurfsmuster ist die Entkopplung von Benutzeransicht, der Applikationslogik und den Anwendungsdaten, vgl. Dustdar u. a. (2003). Im Vergleich zu anderen Web-Frameworks wie Spring Roo oder ASP.NET MVC eignet sich das Play-Framework für kleine Applikationen, da der Aufwand für die Installation und Konfiguration überschaubar ist.

Der Hauptgrund für die Entwicklung einer Web-Applikation zur Verwaltung und Steuerung des Living Place liegt in der Heterogenität der Umgebung. Im Living Place Hamburg kommen verschiedene Betriebssysteme und mobile Plattformen zum Einsatz, für die alle eine Möglichkeit bestehen soll, auf dieses System zuzugreifen. Da es im Rahmen dieser Masterarbeit nicht möglich ist für jedes System eine eigene Applikation zu schreiben, bietet sich diese Lösung an.

Aufbau

Im Folgenden soll ein kurzer Einblick in den Aufbau und die Funktionen der Living Place Management Applikation gegeben werden. Dieser Einblick beschränkt sich auf die Hauptfunktionalitäten. Eine detailliertere Darstellung ist in Anhang B zu finden.

Hauptansicht

Nach der Anmeldung findet sich der Anwender auf der Hauptseite der Web-Applikation wieder. Diese ist in zwei Kategorien aufgeteilt, siehe 4.13. Die obere Kategorie mit dem Namen *User Control Management* beinhaltet die Verwaltung der User, Gruppen und Ressourcen. Je nach Berechtigung des Anwenders sind diese Funktionen sichtbar oder nicht. In diesem Fall besitzt der Anwender Administrationsrechte und hat somit Zugriff auf alle Funktionen, während ein normaler Anwender lediglich die Funktion zur Bearbeitung seiner Profilinformationen, wie Änderung des Passworts oder des Benutzerfotos, etc. besitzt.

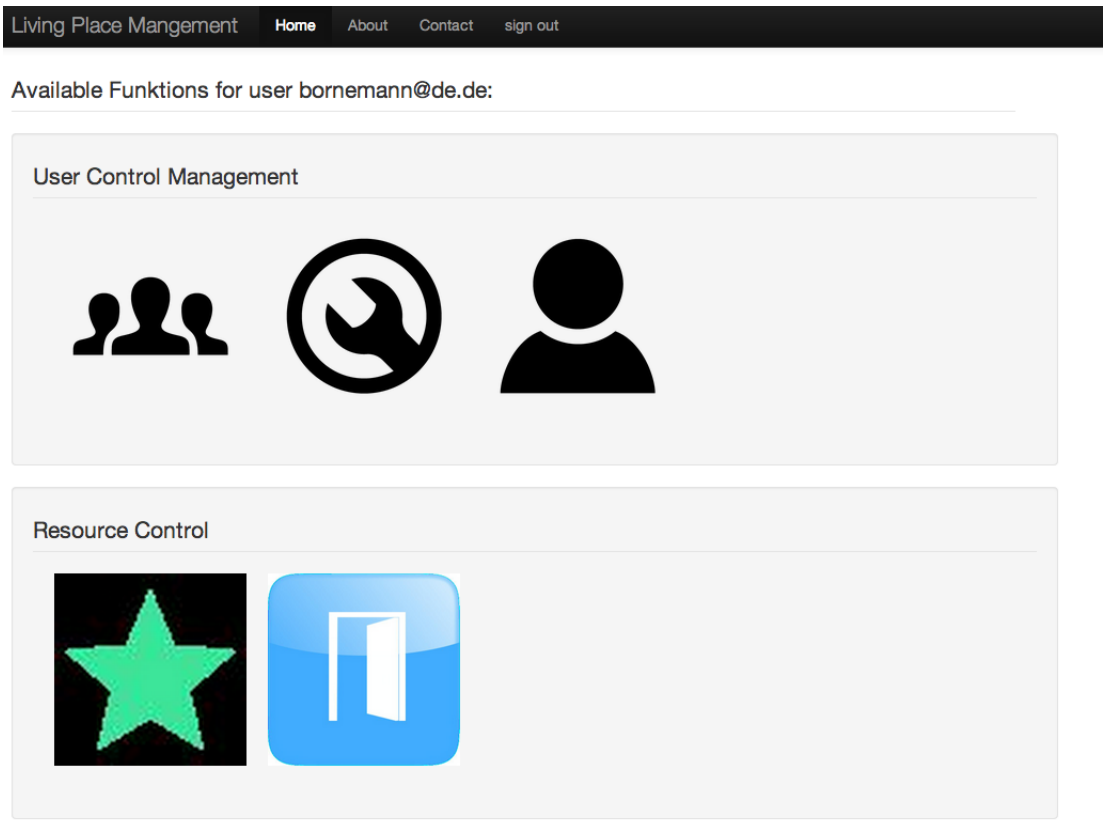

Abbildung 4.13.: Hauptansicht des Living Place Management

Die zweite Kategorie besitzt den Namen *Resource Control*. In diesem Bereich findet der Anwender jene Ressourcen für die er Zugriffsberechtigungen besitzt. Dieser Bereich ist in der Zukunft zur Steuerung der verschiedenen Ressourcen des Living Place Hamburg gedacht und soll somit die bisher verwendete Smartphone-Applikation ablösen. Im Moment verbirgt sich

hinter dem Bereich Resource Control noch keinerlei Funktion. Es werden lediglich die Icons der Ressourcen angezeigt, für die der Anwender Berechtigungen besitzt.

Benutzeransicht

Mit einem Klick auf das entsprechende Symbol, in diesem Fall das User-Symbol, wird eine Seite geöffnet, in der sämtliche Eigenschaften des Anwenders angezeigt werden. In Abbildung 4.14 sind die Eigenschaften eines Anwenders dargestellt. Die Informationen zu einem Benutzer kann man abrufen, indem man in der obersten Dropdown-Box einen Anwender auswählt und den Load-Button betätigt. Ein normaler Benutzer sieht in dieser Box nur seinen eigenen Namen, während ein Anwender mit Administratorrechten jeden User des Systems in der Box aufrufen kann.

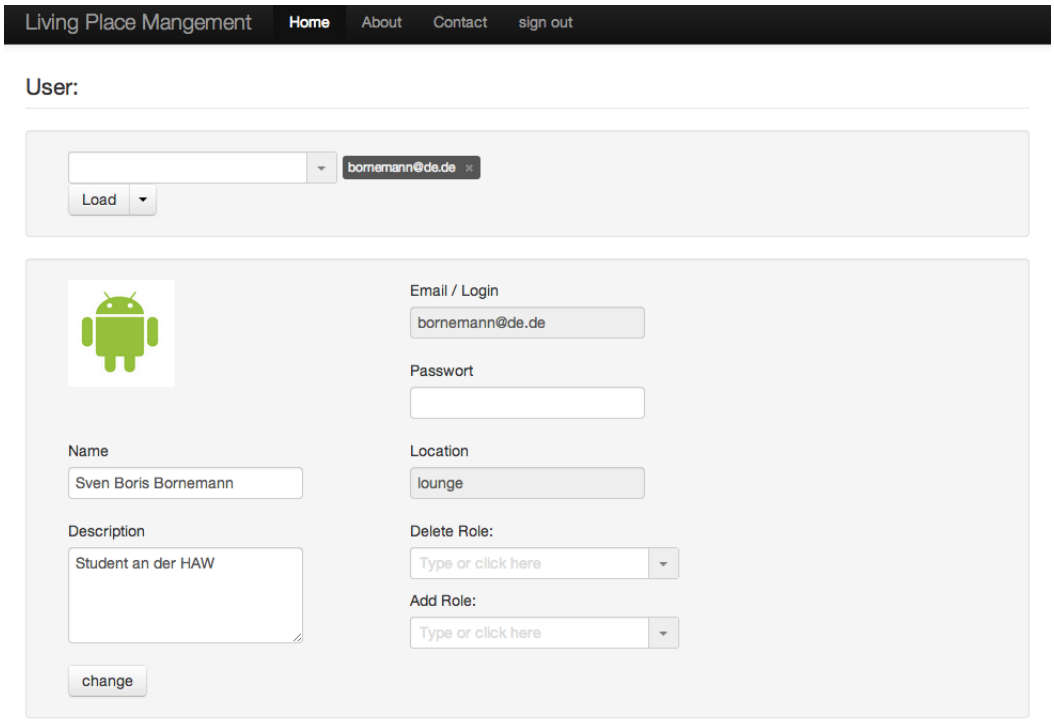

Abbildung 4.14.: Benutzeransicht des Living Place Management

Im Bereich unter der Benutzerauswahl werden dann die Informationen des Benutzers angezeigt. Hierzu zählen beispielsweise das Profilbild, die letzte bekannte Position im Living Place oder verfügbare Gruppen, die zum Profil hinzugefügt oder entfernt werden können.
Das Löschen von Anwendern ist den Benutzern mit Administratorrechten vorbehalten. Dafür

können in der oberen Dropdown-Box mehrere Anwender ausgewählt werden. Anschließend können diese mit einem Klick auf den Pfeil neben dem Load-Button und der anschließenden Auswahl der Löschfunktion aus dem System entfernt werden.
Analog zu diesem Aufbau sind auch die Seiten zur Verwaltung der Gruppen und Ressourcen aufgebaut. Daher wird in diesem Fall auf die Darstellung verzichtet. Eine Darstellung der Seiten ist in Anhang B zu finden.

Kommunikation

Nachdem die Funktionsweise der Oberfläche beschrieben wurde, geht dieser Abschnitt auf die Kommunikation zwischen der Web-Applikation und den anderen Komponenten ein. Exemplarisch wird dies am Vorgang einer Registrierung verdeutlicht. Dazu gibt der Anwender in der Registrierungsmaske alle erforderlichen Daten ein und wird zum System hinzugefügt. Darauffolgend kann er sich am System anmelden. Da dem Benutzer direkt nach der Anmeldung keinerlei Berechtigungen zugewiesen sind, hat dieser nur Zugriff auf die Daten seines Profils. Erst nach der Vergabe von Berechtigungen durch einen Administrator hat der Anwender Zugriff auf Ressourcen.
Der Ablauf einer Registrierung ist im Sequenzdiagramm 4.15 dargestellt. Nach Eingabe der Daten, werden die Informationen über den ActiveMQ an den Home Agent gesendet. Dieser prüft, ob die Person unter diesen Daten bereits im System vorhanden ist. Ist dies nicht der Fall, wird der Anwender mit seinen Daten angelegt und die Web-Applikation leitet den Benutzer zur Login-Seite um.

4.5.2.4. Smart Home Applikation

Die Smart Home Applikation ist im Zuge des Masterstudiums entstanden. Grund hierfür war die Weiterentwicklung einer auf Android basierenden Türklingel Applikation, welche im Rahmen der Bachelorarbeit (vgl. Bornemann (2011)) entstanden ist. Die Idee dahinter bestand darin, die Türklingel in Häusereingängen überflüssig werden zu lassen, in dem diese durch eine mobile Smartphone Applikation ersetzt wird. Des Weiteren sollte eine Zutrittskontrolle für Häuser und Wohnungen realisiert und in die Applikation integriert werden. Wie bereits zuvor erwähnt, sollen die Funktionalitäten der Anwendung in die Web-Applikation User Management übergehen, da diese plattformübergreifend genutzt werden kann. Bisher ist die Nutzung auf Geräte mit einem Android Betriebssystem begrenzt.
Bevor die Funktionen der Applikation genutzt werden können, muss das Smartphone in das Netzwerk der intelligenten Wohnung integriert werden, da ansonsten keine Verbindung zum Berechtigungssystem oder den Services aufgebaut werden kann. Die notwendigen Daten

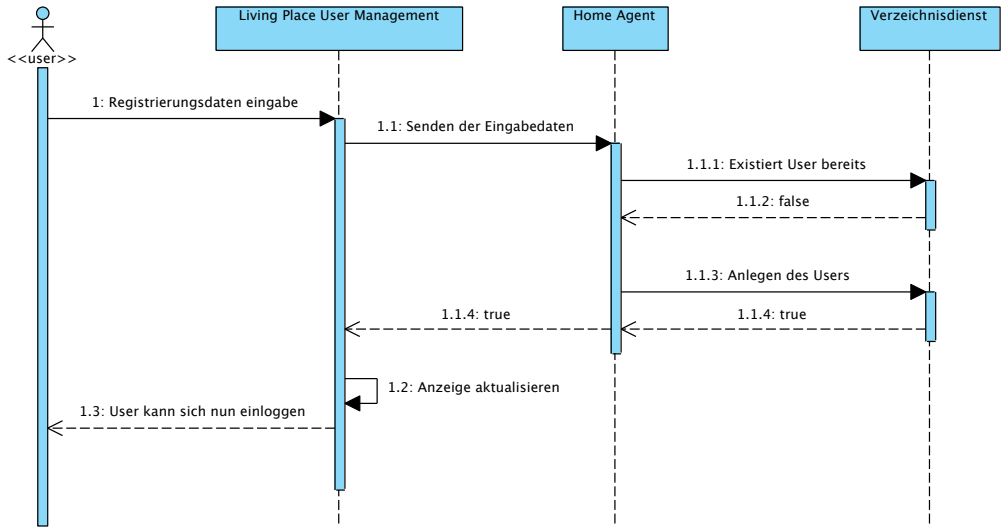

Abbildung 4.15.: Sequenzdiagramm: Registrierung eines neuen Anwenders am Living Place User Management

die das Smartphone benötigt, werden mittels eines NFC-TAGs an der Wohnungstür auf das Smartphone übertragen. Die mit AES (Advanced Encryption Standard) verschlüsselten Daten werden durch die Applikation entschlüsselt und zur Einwahl in das WLAN-Netzwerk der Wohnung verwendet.

Benutzersicht

Die Anwendung ist in zwei Bereiche eingeteilt. Dies ist zum einen der öffentliche Teil der es Personen erlaubt, an einer Wohnung oder einem Haus zu klingeln oder im Falle einer Abwesenheit des Anwohners eine Videonachricht zu hinterlassen. Jene Funktionalitäten wurden aus der Türklingel Applikation übernommen, vgl. Bornemann (2011).
Zum anderen der private Bereich. In diesen Bereich gelangt man nur mit gültigen Zugangsdaten aus dem User Management 4.5.2.3. Mit einem Klick auf das Schlosssymbol werden die Daten an den Home Agent 4.5.2.1 übertragen. Stimmen die Zugangsdaten, werden jene Ressourcen angezeigt, auf die der User Zugriff hat. Betätigt er eine davon, beispielsweise das Symbol zur Öffnung der Tür, werden die ausführbaren Berechtigungen angezeigt. Mit einem Klick auf die Funktion wird dann der Befehl ausgeführt. Der Ablauf der drei eben beschriebenen Schritte ist in Abbildung 4.16 von links nach rechts dargestellt.

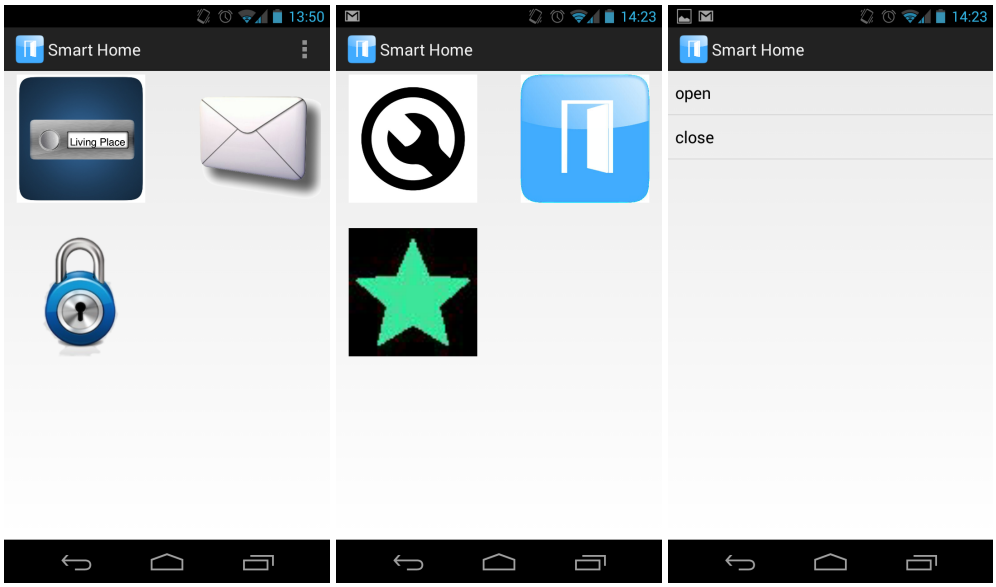

Abbildung 4.16.: Smart Home Applikation: Abrufen der Berechtigungen zur Öffnung der Wohnungstür (Links: Startbildschirm, Mitte: Ressourcen, für die Berechtigungen vorhanden sind, Rechts: Anzeige der verfügbaren Berechtigungen für die Türsteuerung)

Kommunikation

Um den zuvor geschilderten Ablauf zur Ansteuerung der elektrischen Türsteuerung zu ermöglichen, bedarf es der Kommunikation zwischen verschiedenen Komponenten. Im Sequenzdiagramm 4.17 ist diese Kommunikation dargestellt.
Die Login-Daten des Anwenders sendet die Anwendung zum Home Agent. Hier werden die Daten überprüft. Sind die Daten korrekt ,werden alle Berechtigungen des Benutzers ermittelt. Anschließend werden die Informationen an die Smart Home Applikation zurück übermittelt. Der Anwender möchte nun seine Wohnungstür öffnen. Mit dem Absenden dieses Befehls an den Home Agent, wird die in 4.4.3 vorgestellte Regel des CEP-Systems aktiviert. Ist nun die in der Faktenbasis vorhandene Position der Person korrekt, wird der Steuerungsbefehl zur Türsteuerung übermittelt.

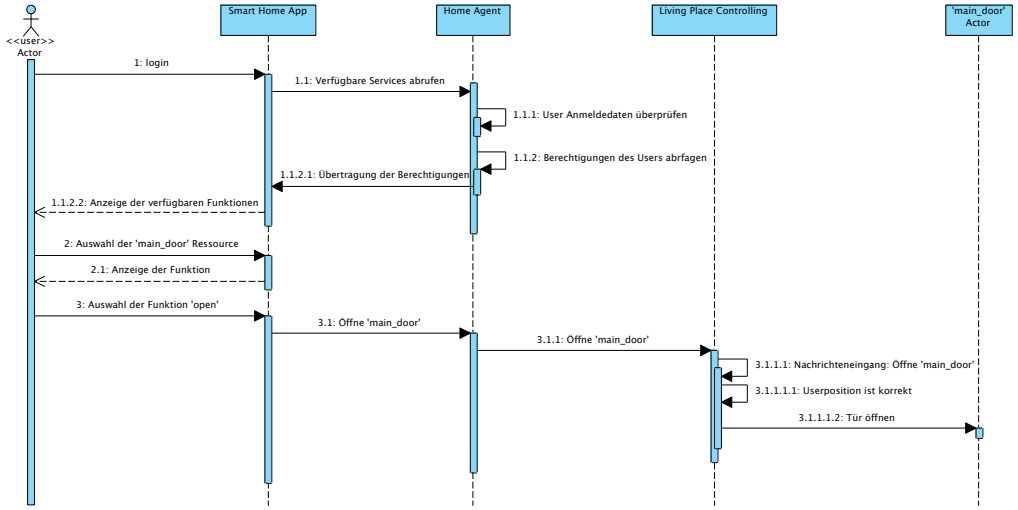

Abbildung 4.17.: Sequenzdiagram: Ansteuerung der elektronischen Türsteuerung

4.5.3. Evaluierung

Nachdem alle Teilkomponenten des Systems vorgestellt wurden, soll in der Evaluierung die korrekte Funktionsweise der Komponenten überprüft werden. Um das Zusammenwirken verschiedener Komponenten zu demonstrieren und zu überprüfen, eignet sich die Öffnung der Wohnungstür aus dem Szenario 3.1.1 sehr gut. Hierfür müssen vier Komponenten miteinander interagieren. Dazu gehören die Berechtigungsstruktur, das CEP-System, die Smart Home Applikation auf dem Smartphone und die elektronische Türsteuerung. Auf die ersten drei Komponenten wurde in dieser Arbeit zuvor eingegangen. Die elektronische Türsteuerung wurde im Rahmen der Projektarbeit des Masterstudiums entwickelt, vgl. Bornemann (2013). Zur Ansteuerung der Tür wurde ein Funk-Türschloss der Firma HomeTec verwendet. Dies zeichnet sich vor allem durch seine unkomplizierte Montage aus, da es lediglich auf dem Türschloss angebracht werden muss. Das Funk-Türschloss besitzt einen Motor, welcher durch eine Funkfernbedienung den Schlüssel im Schloss in die gewünschte Richtung dreht und so die Tür öffnet oder verschließt.
Um das Funk-Türschloss in das System zu integrieren, muss eine Schnittstelle geschaffen werden, welche die Steuerung des elektronischen Schlosses über eine Ethernet-Schnittstelle ermöglicht. Hierfür wurde die Fernsteuerung mit einem Raspberry Pi verbunden, siehe Abbildung 4.18. Des Weiteren wurde ein Programm geschrieben welches Befehle für das Funk-Türschloss über den ActiveMQ empfangen kann und diese in elektrische Signale für die Fernsteuerung umsetzt. Somit kann das Schloss von den Komponenten des Living Place angesteuert werden.

Zur Öffnung der Tür sind nun alle Komponenten vorhanden und die Ansteuerung kann über

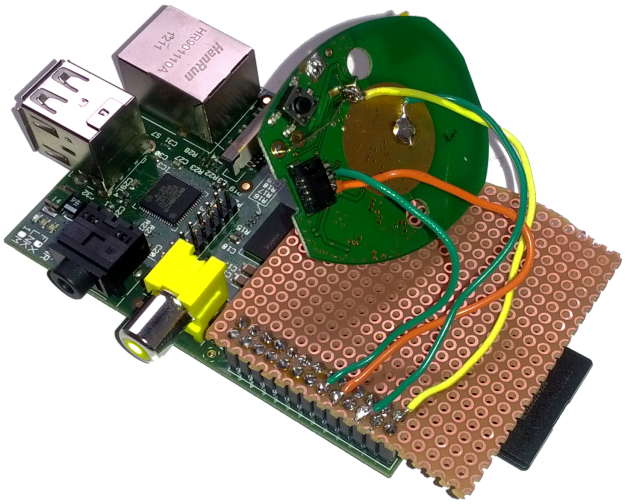

Abbildung 4.18.: Raspberry Pi verbunden mit der Fernsteuerung des Funk-Türschlosses

die Smart Home Applikation erfolgen. Das Indoor-Positioning System des Living Place publiziert jede Positionsänderung einer Person auf dem ActiveMQ. Diese werden vom CEP-System erfasst und in der Faktenbasis abgelegt. Eine berechtigte Person kann mittels der Applikation den Befehl zum Öffnen der Tür absenden. Das CEP-System gleicht den Befehl mit der Position der Person ab. Befindet sich die Person in der Nähe der Wohnungstür, wird der Befehl über den ActiveMQ an den Raspberry Pi weitergeleitet. Dieser steuert die Fernsteuerung des Schlosses an, worauf diese die Tür öffnet. 30 Sekunden später wird die Position der Person aus der Faktenbasis gelöscht. Somit wird gewährleistet, dass die Person die Tür erst wieder öffnen kann, wenn sie sich ihr nähert. Abbildung 4.19 zeigt exemplarisch das eben beschrieben Vorgehen zur Öffnung der Wohnungstür durch den Anwender.

Der eben beschriebene Ablauf hat gezeigt, dass die gewünschten Funktionen realisiert wurden und die Komponenten einwandfrei miteinander kommunizieren. Ziel war es, die Berechtigung zum Öffnen einer Wohnungstür an einen bestimmten Kontext zu binden. In diesem Fall die Position einer Person. Wenn in Zukunft weitere Umgebungsinformationen in der Faktenbasis abgelegt werden, können etliche Funktionalitäten im Living Place Hamburg, abhängig vom Kontext, ausgeführt werden. Funktionen und Abläufe können automatisiert und in Abhängigkeit eines Kontextes ausgeführt werden, ohne dass eine explizite Interaktionen des Bewohners erforderlich ist.

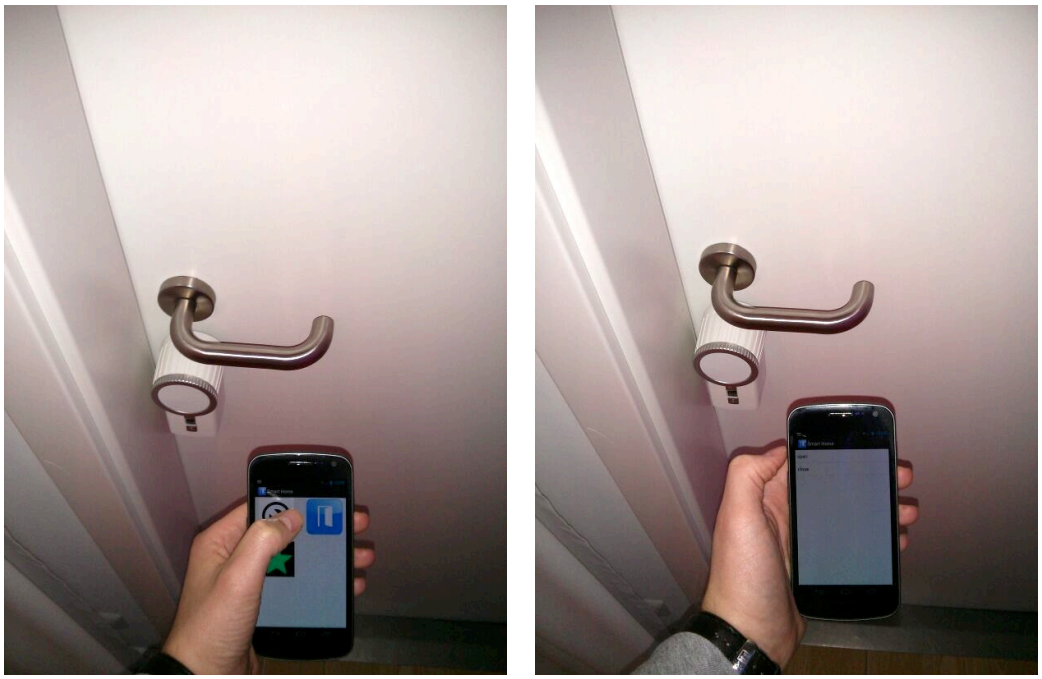

Abbildung 4.19.: Smart Home Applikation (Links: Auswahl der zu steuernden Ressource, Rechts: Auswahl der gewünschten Funktion)

4.6. Fazit

Im vergangenen Designkapitel wurde eine komplexe Architektur und deren Realisierung vorgestellt. Diese Realisierung erfüllt die im Fazit der Analyse 3.6 identifizierten Anforderungen. Die Ablage der erforderlichen Daten wird durch die vorgestellte Berechtigungsstruktur realisiert. Infolge der erstellten Schemata ist es möglich jedem Anwender nicht nur Zugriff auf alle Funktionen einer Ressource zu gewähren, sondern den Zugriff auf einzelne Funktionen zu verfeinern. Anderen Lösungen, wie beispielsweise die Entwicklung einer eigenen Datenbankanwendung, sind für die Realisierung dieser Funktionen ebenfalls möglich. Das verwendete LDAP-Protokoll eignet sich hier aber besonders, da durch die interne Baumstruktur, die Daten schnell abgefragt werden können. Des Weiteren sind Mechanismen zur Autorisierung sowie Authentifizierung vorhanden und können genutzt werden.
Mit der Verwendung und Integration des CEP-Systems Drools Fusion, ist die Erkennung des Kontextes und die Reaktion auf diesen möglich. Die Funktionsweise des Systems wurde durch die Implementierung und Durchführung des Szenarios: Familienurlaub 3.1.1 geprüft. Somit ist die Kontextsensitivität des Systems gewährleistet.

Aufgrund einiger Hindernisse konnten die einzelnen Komponenten des Systems nicht wie geplant als OSGi-Bundles implementiert werden. Dies sollte einer der nächsten Schritte bei der Erweiterung des Systems sein. Durch diesen Schritt würde die Wartung und ebenso die Bedienung der einzelnen Komponenten erleichtert.
Aufgrund des Zeitrahmens, der für diese Arbeit zur Verfügung stand, konnten nur rudimentäre Funktionalitäten zur Verfügung gestellt werden. Die Erstellung weiterer Sensoren, zur Gewinnung von Kontextinformationen und Aktoren, zur Reaktion auf den aktuellen Kontext, wäre wünschenswert gewesen. Des Weiteren ist das Hinzufügen und Entfernen von Regeln zur Laufzeit des Systems nicht ohne Weiteres möglich. Dies bedeutet, das System muss bei einer Regelanpassung neu gestartet werden. Ob sich Drools Fusion dennoch für einen Produktiveinsatz eignet oder ein anderes CEP-System verwendet werden sollte, muss in Zukunft noch evaluiert werden.
Durch einen größeren Zeitrahmen und mehreren Personen im Projekt wären die genannten Punkte durchaus realisierbar gewesen. Hierdurch hätten auch Verfahren zur Erstellung, Bearbeitung und Zuweisung von Berechtigungen durch die Integration intelligenter Lernverfahren unterstützt werden können.

Zukünftige Arbeiten in diesem Bereich können dieses System als Ausgangspunkt nutzen

und die eben genanten Funktionalitäten im Rahmen ihrer Projekt- oder Abschlussarbeiten hinzufügen. Des Weiteren können durch die Modularisierung der Komponenten und dem gewählten Kommunikationsmechanismus weitere Systeme angebunden oder Komponenten ausgetauscht werden. Um beispielsweise Abrechnungen für gewisse Services zu ermöglichen, könnte ein Abrechnungssystem integriert werden. Sinnvoll könnte dies im vorgestellten Szenario Parkplatz-Sharing 3.1.4 sein. Die auftretenden Parkplatzgebühren könnten so unmittelbar im System abgerechnet werden.

5. Integration in das Living Place Hamburg

Während in den vorherigen Kapiteln die Entwicklung der Komponenten im Vordergrund stand, beschäftigt sich dieses Kapitel mit der Integration dieser Komponenten in das Living Place Hamburg.
Als Basis fungiert ein virtuelle Maschine, welche mit der Linux Distribution Debian 6 ausgestattet wurde. Auf dieser wurden allen notwendigen Paketen installiert. Hierzu gehören unter anderem das aktuelle Java Development Kit in der Version 7, der ApacheDS und der Open Source Webserver Nginx. Der Nginx Webserver wurde ausgewählt, da dieser leicht zu konfigurieren ist, wenig Ressourcen benötigt und dabei trotzdem eine gute Performance bietet, vgl. Nginx. Der Webserver wurde so konfiguriert, dass dieser die erstellte Web Applikation User Management unter der Adresse *https://lpusermanagement.de* zur Verfügung stellt. Die benötigten Zertifikate zum Aufbau einer HTTPS-Verbindung wurden mit *openssl* erstellt und in die Konfiguration des Webservers eingearbeitet. Die Konfiguration ist im Listing 5.1 zu finden.

```
1  user  nginx;
2  worker_processes  1;
3
4  error_log  /var/log/nginx/error.log warn;
5  pid        /var/run/nginx.pid;
6
7  events {
8      worker_connections  1024;
9  }
10
11 http {
12     include       /etc/nginx/mime.types;
13     default_type  application/octet-stream;
14
15     log_format  main  '$remote_addr - $remote_user [$time_local]
           "$request" '
16                       '$status $body_bytes_sent "$http_referer" '
```

```
                    '"$http_user_agent" "$http_x_forwarded_for
                       "';

    access_log  /var/log/nginx/access.log  main;

    sendfile        on;

    keepalive_timeout  65;

    include /etc/nginx/conf.d/*.conf;

        proxy_buffering    off;
        proxy_set_header   X-Real-IP $remote_addr;
        proxy_set_header   X-Scheme $scheme;
        proxy_set_header   X-Forwarded-For
           $proxy_add_x_forwarded_for;
        proxy_set_header   Host $http_host;
        proxy_set_header   X-Forwarded-Proto https;
        proxy_set_header   X-Forwarded-Ssl on;

    upstream lpusermanagement {
        server 127.0.0.1:9000;
      }

    server {
        listen                 443 ssl;
        server_name            lpusermanagement.de

        ssl                    on;
        ssl_certificate        /etc/ssl/lpusercerts/lp_server.crt;
        ssl_certificate_key    /etc/ssl/lpusercerts/lp_server.key;
        ssl_session_timeout    5m;


        location / {
                proxy_pass    http://lpusermanagement;
                proxy_set_header Host $http_host;
           proxy_set_header X-Real-IP $remote_addr;
           proxy_set_header X-Forwarded-Proto https;
```

```
54              proxy_set_header X-Forwarded-For
                    $proxy_add_x_forwarded_for;
55              proxy_redirect http:// https://;
56              add_header Pragma "no-cache";
57          }
58      }
59  }
```

Listing 5.1: Nginx Webserver Konfiguration

Da zum Zeitpunkt der Installation jedoch kein DNS Dienst im Netzwerk des Living Place zur Verfügung stand, kann die User Management Web Applikation nur über die Eingabe der aktuellen IP-Adresse des Servers erreicht werden. Alternativ kann auch ein entsprechender Eintrag in die Host-Datei des jeweiligen Rechners vorgenommen werden.
Nachdem der Server mit allen notwendigen Paketen ausgestattet und diese konfiguriert wurden, findet im folgenden Abschnitt die Inbetriebnahme der Komponenten statt.
Alle vorgestellten Komponenten wurden in das Git-Repository des Living Place Hamburg hochgeladen. Somit können Studenten zu einem späteren Zeitpunkt auf den Implementationsstand der Komponenten zugreifen.

5.1. Inbetriebnahme der Komponenten

Im Zuge der Masterarbeit wurden alle Komponenten auf demselben Server in Betrieb genommen. Aufgrund der gewählten Kommunikationsmechanismen können diese aber auch auf unterschiedlichen Servern in einem Netzwerk eingesetzt werden.
Der ApacheDS, der Home Agent sowie das User Management wurden so konfiguriert, dass sie beim Hochfahren des Servers automatisch gestartet werden. Somit ist das System nach einem Stromausfall oder einem Neustart sofort einsatzbereit und bedarf keiner weiteren Steuerung durch einen Endanwender.

Verzeichnisdienst

Als Verzeichnisdienst wurde der ApacheDS in der Version 2M11 auf dem Server installiert und zu dem Systemstart des Servers zugewiesen. Im nach der Installation des Verzeichnisdienstes, wurde dieser um eine neue Partition mit dem Namen *dc=livingplace,dc=org* erweitert. In dieser Partition wurden die Organisationseinheiten: users, groups und resources eingerichtet und mit exemplarischen Datensätzen gefüllt, siehe Abbildung 5.1. Die restliche Konfiguration des ApacheDS wurde nicht verändert.

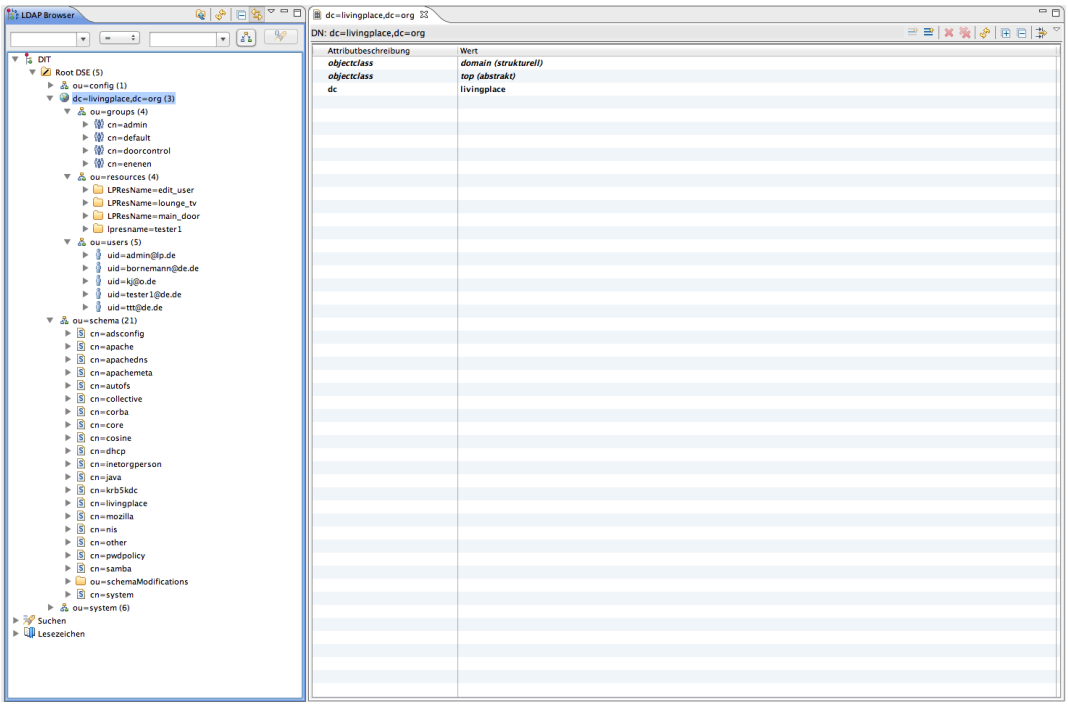

Abbildung 5.1.: Beispielkonfiguration des Verzeichnisdienstes

Home Agent

In Listing 5.2 wird das Startskript des Home Agent dargestellt. Wie aus diesem zu entnehmen ist, wurde das Projekt in eine ausführbare Jar-Datei umgewandelt und im Home-Odner des Benutzers *lpuser* platziert. Das Startskript ist unter */etc/init.d/* zu finden. Damit das Skript beim Start des Systems ausgeführt wird, wurde dies mit dem Befehl *update-rc.d* dem Systemstart hinzugefügt.

```
#!/bin/bash
# Home Agent
# Maintainer: Sven Boris Bornemann @ HAW
# App Version: 1.0

### BEGIN INIT INFO
# Provides: Service for Home Agent
# Required-Start: $local_fs
# Required-Stop: $local_f
# Default-Start: 2 3 4 5
# Default-Stop: 0 1 6
# Short-Description: Home Agent
# Description: Home Agent represents interface between Lp and
   Usermanagement
### END INIT INFO
case $1 in
    start)
        cd /home/lpuser/projects/HomeAgent
        /usr/bin/java -jar homeagent.jar &
    ;;
    stop)
        /bin/bash /etc/init.d/homeagent-stop.sh &
    ;;
    restart)
        /etc/init.d/homeagent-stop.sh
        cd /home/lpuser/projects/HomeAgent
        /usr/bin/java -jar homeagent.jar &
    ;;
esac
exit 0
```

Listing 5.2: Start Skript des Home Agent

User Management

Um das User Management auszuführen wurde ebenfalls ein Startskript erstellt und dem Systemstart hinzugefügt. Der Aufbau ist analog zum vorherigen Skript aus Listing 5.2.
Damit das Play-Framwork nicht auf dem Server installiert werden muss, wurde das User Management Projekt als Standalone Version kompiliert. Hierfür muss im Verzeichnis des Projekts, der Befehl *play clean compile stage* ausgeführt werden. Im Anschluss wurde das kompilierte Projekt auf den Server in das Home-Verzeichnis des *lpuser* übertragen. Das Startskript und einen Überblick der User Management Applikation sind in Anhang B zu finden.

6. Schluss

Im letzten Kapitel dieser Arbeit wir zunächst eine Zusammenfassung 6.1 über die Vorgehensweise in dieser Arbeit gegeben. Im Anschluss daran werden die gewonnenen Kenntnisse aus dieser Arbeit sowie mögliche Erweiterungen für die zukünftige Entwicklung des Systems vorgestellt, vgl. Abschnitt 6.2.

6.1. Zusammenfassung

In der Arbeit wurde der Ansatz und die Realisierung für ein auf Wohnumgebungen zugeschnittenes Berechtigungssystems vorgestellt. Dieses zeichnet sich durch die Verwendung dynamischer Berechtigungen aus, welche eine Zugriffskontrolle auf Ressourcen einer Wohnumgebung in Abhängigkeit zum aktuellen Kontext ermöglichen. Das entstandene, kontextsensitive Berechtigungssystem besteht aus mehreren einzelnen Komponenten, welche die Funktionalitäten des Systems zur Verfügung stellen.
Im Analysekapitel 3 wurden die Anforderungen an das erstellte System ermittelt. Hierzu wurden vier Szenarien vorgestellt, die alltägliche Situationen einer fiktiven Familie widerspiegelten. Die Szenarien wurden hinsichtlich ihrer Umgebungsinformationen untersucht. Anhand dieser wurden Kontextinformationen abgeleitet, welche zur Identifizierung des Kontextes dienen. Im folgenden Designkapitel 4 wurde ein Informationsmodell zur Speicherung der Umgebungsdaten entwickelt und vorgestellt. Die Strukturierung des Modells basiert auf den gesammelte Information der Umgebungsanalyse 3.3. Mit dem Aufbau des Verzeichnisdiensts wurde der Grundstein für das Berechtigungssystem gelegt. Statische Berechtigungen und die Authentifizierung von Personen konnte darüber abgehandelt werden. Im Anschluss wurde das Complex Event Processing System vorgestellt, vgl. Abschnitt 4.3. Diese Komponente ermöglicht die Erfassung von Kontextinformationen. Die vorhanden Informationen können durch Regeln verknüpft und zur Kontextidentifizierung genutzt werden. Des Weiteren wird durch die Regeln definiert, welche Reaktionen ausgeführt werden, wenn der entsprechende Kontext zutrifft. Mit der Verwendung des CEP-Systems kann das Berechtigungssystem auf aktuelle Geschehnisse reagieren und ist somit kontextsensitiv.

Zudem wurde im Abschnitt 4.5 die Realisierung des Systems beschrieben. Dort wurde zunächst ein Gesamtüberblick des Systems gegeben. Darin wird die Kommunikation zwischen den Komponenten verdeutlicht. Des Weiteren werden die Funktionalitäten der einzelnen Module beschrieben. Im Anschluss an den Gesamtüberblick folgt eine detailliertere Vorstellung der einzelnen Komponenten.
Im Kapitel 5 wird beschrieben, wie die Komponenten in das Living Place Hamburg integriert wurden. Hierzu gehört, auf welchem System die Module laufen und wie dieses eingerichtet ist. Des Weiteren werden die verschiedenen Konfigurationen der einzelnen Komponenten beschrieben.

6.2. Fazit und Ausblick

Die Zugriffskontrolle auf Ressourcen in der Geschäftswelt sind gang und gäbe. Ziel dieser Arbeit war es diese Mechanismen auch für das private Umfeld zur Verfügung zu stellen, um den Bewohnern zukünftiger Smart Homes eine Möglichkeit der Kontrolle anzubieten. Eine Kontrolle darüber, welche Zugriffe auf Ressourcen erlaubt sind und wer diese ausführen darf. Dabei herrschen im privaten Umfeld anderen Anforderungen als in der Geschäftswelt. Berechtigungen können nicht einfach anhand der Zugehörigkeit oder dem Stand einer Person, wie beispielsweise Abteilung und Posten im Unternehmen, festgemacht werden. Im privaten Umfeld werden Berechtigungen unter anderem durch soziale Hierarchien festgelegt. Des Weiteren sind Berechtigungen meist komplexer als ein simpler Zugriff auf eine Ressource. Aus diesem Grund müssen Informationen der Umgebung mit in das System einfließen. Erst durch diese Daten können die komplexen Berechtigungen realer Situationen abgebildet werden.
Ziel dieser Arbeit war die Entwicklung eines kontextsensitiven Berechtigungssystems für Smart Homes und dessen Integration in das Living Place Hamburg. Zum einen soll das System den Bewohner eines Smart Homes ermöglichen die Zugriffe auf Ressourcen zu steuern. Zum anderen soll die Sicherheit in intelligenten Wohnumgebungen gesteigert werden. Mit Sicherheit ist in diesem Fall die Zugriffsbeschränkung auf Ressourcen gemeint, vor allem kann dadurch die Verbreitung privater Daten und Informationen verhindert werden.
Ob das System bei den Menschen Akzeptanz findet, hängt von verschiedenen Faktoren ab. Hierzu zählen menschliche sowie technische Faktoren. Wie werden Menschen reagieren, wenn ihnen in ihrer Wohnung Zugriffe auf Geräte verwehrt bleiben, die sie vorher selbstverständlich genutzt haben? Wie kommen ältere Menschen mit dem System klar und akzeptieren diese einen Rechteentzug? Das sind Fragen die in Zukunft beantwortet werden müssen.
Nicht nur die Funktionalitäten sind ein wichtiger Grund für Akzeptanz. Auch der Spaß, die

Aufregung und Freude die ein Mensch mit dem Umgang des Systems empfindet sind relevant. Ebenfalls spielt hier das Design eine Rolle, vgl. Norman (2004).
Ob das System technisch überzeugen kann hängt vor allem davon ab, ob das System dem Menschen einen Mehrwert geniert. Auch ist entscheidend, ob die Anwender gute Erfahrungen mit dem System verbinden. Häufiges Fehlverhalten des Systems wird in einem sensiblen Umfeld, wie der eigenen Wohnumgebung, zu einer Abneigung führen. Auch die Anpassungsmöglichkeit an die Wünsche des Menschen ist ein Kriterium. Die Wohnumgebung eines Menschen stellt eine komplexe Umgebung dar, in der jeder Mensch andere Wünsche und Bedürfnisse besitzt.

Potential für zukünftige Weiterentwicklungen des Systems sind vorhanden. So können diverse Automatisierungen im Bereich der Nutzerverwaltung, Rechtezuweisung und Ressourcenintegration vorgenommen werden. Dem Anwender wird somit der Umgang mit dem System erleichtert. In diesem Zusammenhang muss jedoch auch erforscht werden, welche Automatisierungen sinnvoll sind, welche Eingriffsmöglichkeiten dem Anwender geboten werden und in welcher Form der Benutzer über Abläufe des Systems informiert wird. Bei einer falschen Balance dieser Aspekte könnte ansonsten eine Überforderungen beziehungsweise Hilflosigkeit gegenüber dem System entstehen.
Durch vorhanden Daten im System kann dieses um zusätzliche Funktionen erweitert werden. So wäre beispielsweise die Realisierung eines Service Discovery Funktionalität eine sinnvolle Erweiterung, um die Integration weiterer Services zu vereinfachen. Des Weiteren kann durch die Verschlüsselung der Daten sowie der gesamten Kommunikation, die Sicherheit des Systems weiter gesteigert werden.
Aufgrund des flexiblen Aufbaus kann das System beliebig angepasst und auf die individuellen Bedürfnisse angepasst werden. Dies schließt nicht nur Services einer Wohnumgebung mit ein, sondern auch Services anderer Smart Environments.

A. Home Agent

Im Folgenden werden weiterführende Informationen des Home Agent präsentiert. Hierzu gehört das in Abbildung A.1 gezeigt Klassendiagramm. Anhand der Klassen und Methoden, soll die Funktionsweise des Home Agent weiter verdeutlicht werden.

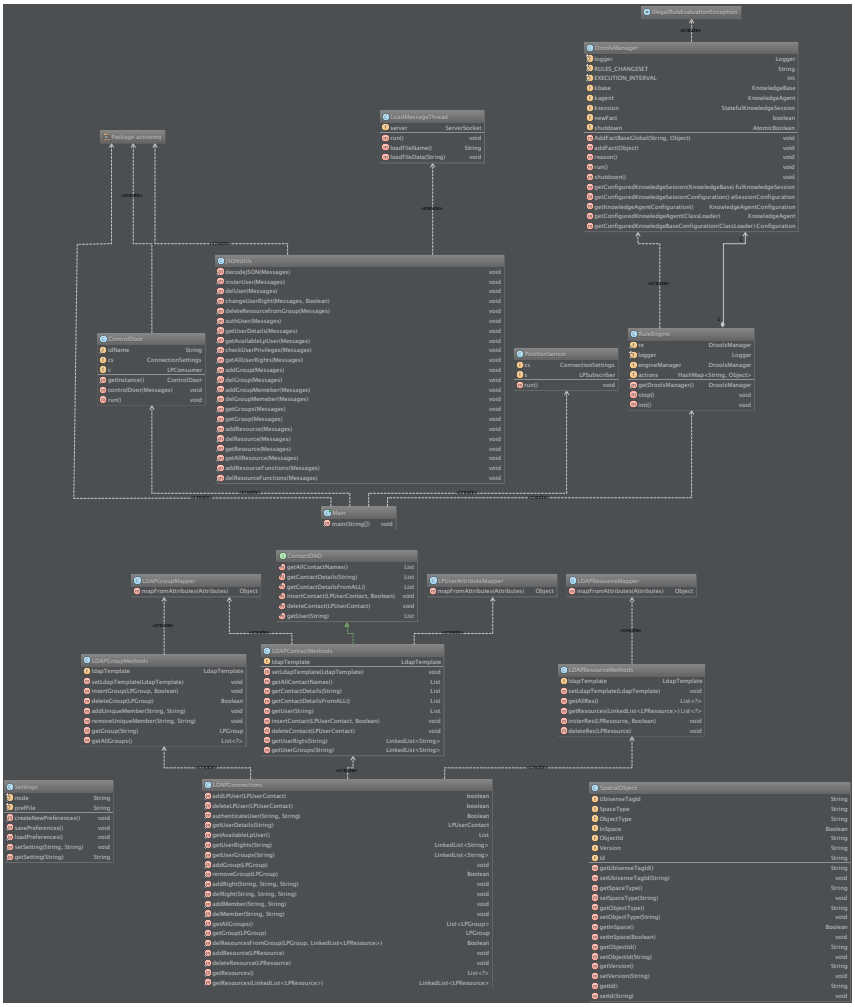

Abbildung A.1.: Klassendiagramm: Home Agent

B. User Management

Wie auch beim Home Agent sollen hier detaillierter Informationen zum Aufbau sowie den Funktionalitäten des User Management bereitgestellt werden. Dafür werden zunächst weitere Möglichkeiten zur Steuerung des Systems durch den Anwender präsentiert. Anschließend folgt eine Grafik zum strukturellen Aufbau der Anwendung. Zum Schluss werden die Skripte zum Starten beziehungsweise Stoppen der Anwendung aufgelistet.

B.1. Anwendersicht

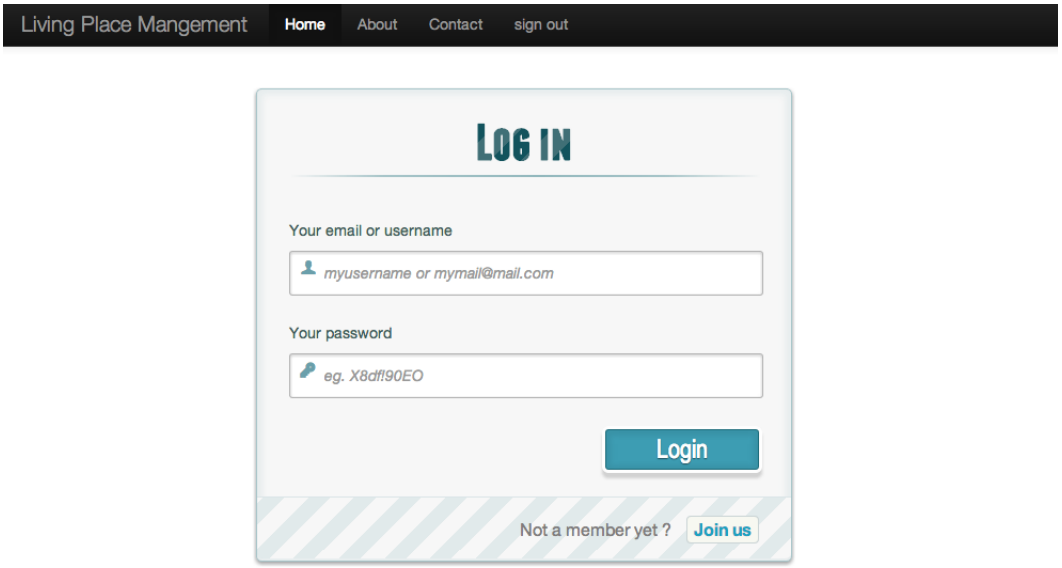

Abbildung B.1.: Anmeldeansicht des Living Place Management

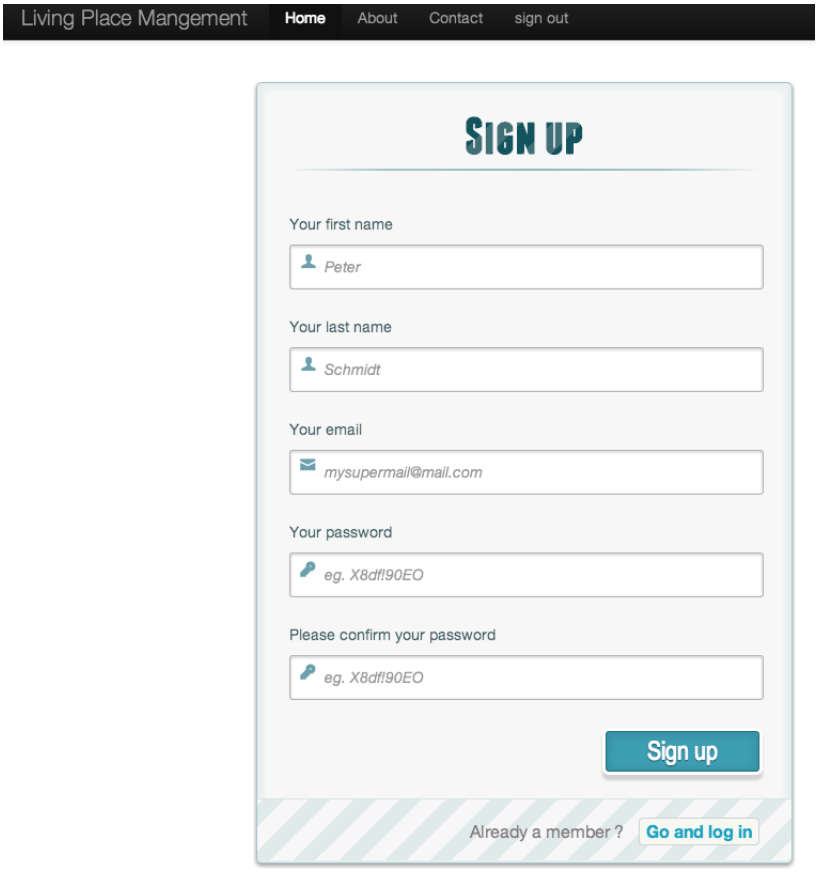

Abbildung B.2.: Registrierungsansicht für neue Anwender des Living Place Management

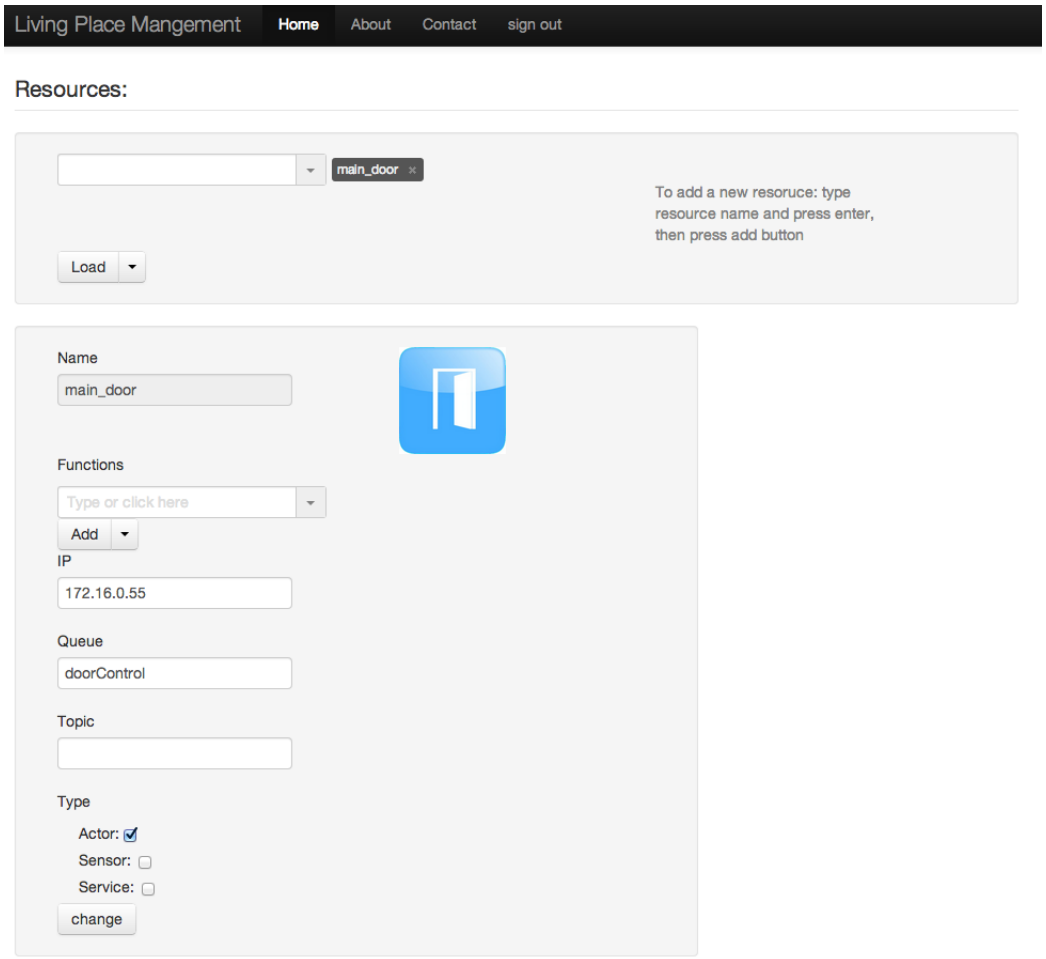

Abbildung B.3.: Ressourcenansicht des Living Place Management

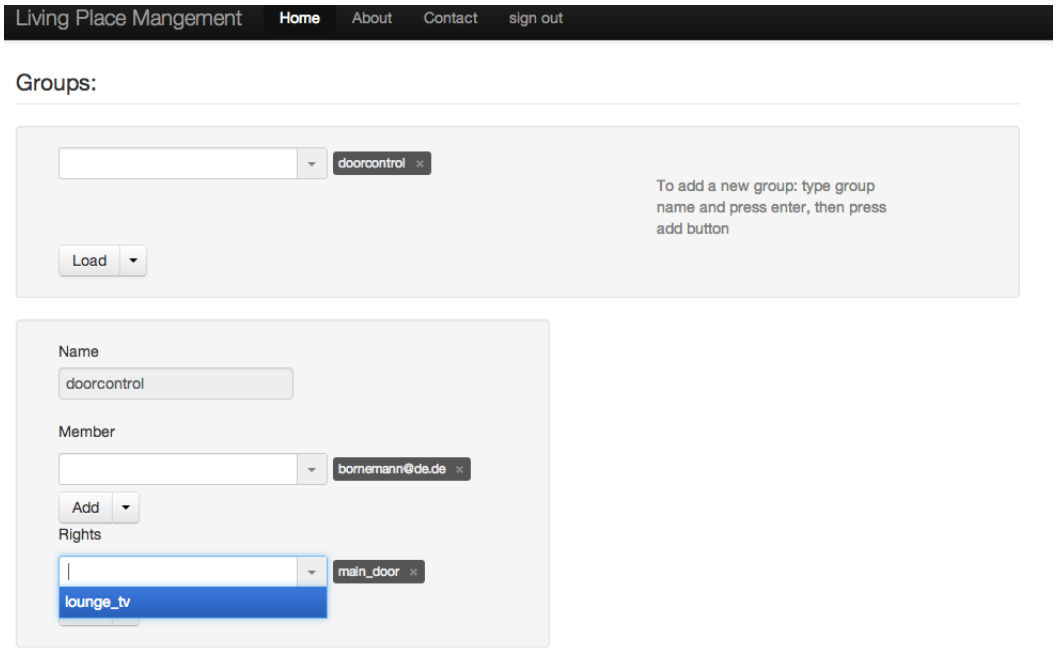

Abbildung B.4.: Gruppenansicht des Living Place Management

B.2. Struktur

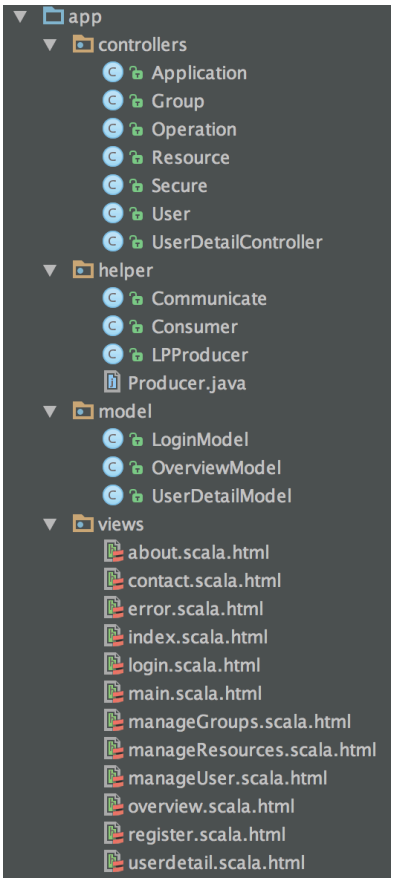

Abbildung B.5.: Struktureller Aufbau des Living Place User Management

B.3. Skripte

```
#!/bin/bash
cd /home/lpuser/projects/LPUserManagement/
./start
```

Listing B.1: User Management Start Skript

```
#!/bin/bash
pid=‘cat /home/lpuser/projects/RUNNING_PID‘
kill -9 $pid
rm /home/lpuser/projects/RUNNING_PID
```

Listing B.2: User Management Stop Skript

```
#!/bin/bash
# LP User Management
# Maintainer: Sven Boris Bornemann @ HAW
# App Version: 1.0

### BEGIN INIT INFO
# Provides: LP User Management Service
# Required-Start: $local_fs
# Required-Stop: $local_f
# Default-Start: 2 3 4 5
# Default-Stop: 0 1 6
# Short-Description: LP User Management
# Description: CLP User Management Website
### END INIT INFO

case $1 in
    start)
       /bin/bash /etc/init.d/lpusermanagement-start.sh &
    ;;
    stop)
        /bin/bash /etc/init.d/lpusermanagement-stop.sh &
    ;;
    restart)
        /bin/bash /etc/init.d/lpusermanagement-stop.sh
        /bin/bash /etc/init.d/lpusermanagement-start.sh &
    ;;
```

```
27 esac
28 exit 0
```

Listing B.3: User Management Skript

Literaturverzeichnis

[Abowd u. a. 1999] Abowd, Gregory D. ; Dey, Anind K. ; Brown, Peter J. ; Davies, Nigel ; Smith, Mark ; Steggles, Pete: Towards a Better Understanding of Context and Context-Awareness. In: *Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing*. London, UK, UK : Springer-Verlag, 1999 (HUC '99), S. 304–307. – URL http://dl.acm.org/citation.cfm?id=647985.743843. – (abgerufen am: 14.06.2013). – ISBN 3-540-66550-1

[Augusto 2007] Augusto, Juan C.: Ambient Intelligence: the Confluence of Ubiquitous/Pervasive Computing and Artificial Intelligence. (2007). – URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.147.553&rep=rep1&type=pdf. – abgerufen am 07.07.2011

[Aware Home] Aware Home: *The Aware Home Research Initiative*. – URL http://www.awarehome.gatech.edu/. – (abgefrufen am 13.06.2013)

[Bornemann 2011] Bornemann, Sven B.: Android-basierte Smart Home Interaktion am Beispiel einer Gegensprechanlage. (2011). – URL http://users.informatik.haw-hamburg.de/~ubicomp/arbeiten/bachelor/bornemann.pdf. – (abgerufen am: 12.10.2013)

[Bornemann 2012] Bornemann, Sven B.: Integration mobiler Endgeräte in Smart Homes mittels NFC. (2012). – URL http://users.informatik.haw-hamburg.de/~ubicomp/projekte/master2012-proj1/bornemann.pdf. – (abgerufen am: 18.10.2013)

[Bornemann 2013] Bornemann, Sven B.: Aufbau einer Berechtigungsstruktur für Smart Homes. (2013). – URL http://users.informatik.haw-hamburg.de/~ubicomp/projekte/master12-13-proj2/bornemann.pdf. – (abgerufen am: 19.09.2013)

[Brown u. a. 1997] Brown, P.J. ; Bovey, J.D. ; Chen, Xian: Context-aware applications: from the laboratory to the marketplace. In: *Personal Communications, IEEE* 4 (1997), Nr. 5, S. 58–64. – ISSN 1070-9916

[Capgemini Consulting 2011] Capgemini Consulting: Smart Home - Zukunftschancen verschiedener Industrien. (2011). – URL http://www.de.capgemini-consulting.com/sites/default/files/resource/pdf/smart_home_-_zukunftschancen_verschiedener_industrien_0.pdf. – (abgerufen am: 02.10.2013)

[Cook und Das 2004] Cook, Diane ; Das, Sajal: *Smart Environments - Technology, Protocols and Applications*. New York : John Wiley & Sons, 2004. – ISBN 978-0-471-68658-3

[Cook u. a. 2003] Cook, D.J. ; Youngblood, M. ; Heierman, III ; Gopalratnam, K. ; Rao, S. ; Litvin, A. ; Khawaja, F.: MavHome: an agent-based smart home. In: *Pervasive Computing and Communications, 2003. (PerCom 2003). Proceedings of the First IEEE International Conference on*, 2003, S. 521–524

[Covington u. a. 2001] Covington, Michael J. ; Long, Wende ; Srinivasan, Srividhya ; Dev, Anind K. ; Ahamad, Mustaque ; Abowd, Gregory D.: Securing context-aware applications using environment roles. In: *Proceedings of the sixth ACM symposium on Access control models and technologies*. New York, NY, USA : ACM, 2001 (SACMAT '01), S. 10–20. – URL http://doi.acm.org/10.1145/373256.373258. – ISBN 1-58113-350-2

[Das u. a. 2002] Das, S.K. ; Cook, D.J. ; Battacharya, A. ; Heierman, III ; Lin, Tze-Yun: The role of prediction algorithms in the MavHome smart home architecture. In: *Wireless Communications, IEEE* 9 (2002), Nr. 6, S. 77–84. – ISSN 1536-1284

[Dey und Abowd] Dey, Anind K. ; Abowd, Gregory D.: *CyberDesk: The Use of Perception in Context-Aware Computing*. Extended abstract presented as a poster in the Proceedings of 1997 Workshop on Perceptual User Interfaces (PUI '97), pages 26-27, October, 1997. – URL http://www.cc.gatech.edu/fce/cyberdesk/pubs/PUI97/pui.html. – (abgerufen am: 14.06.2013)

[Dreschke 2011] Dreschke, Oliver: *Entwicklung kontextsensitiver Möbel für intelligente Wohnumgebungen*, Hochschule für Angewandte Wissenschaften Hamburg, Masterthesis, 2011

[DRK] DRK: *Deutsches Rotes Kreuz - Hausnotruf*. – URL http://www.drk-hausnotruf.net/seite.asp?selTopLevel=1&siteID=48. – (abgerufen am: 14.06.2013)

[Dustdar u. a. 2003] Dustdar, Schahram ; Gall, Harald ; Hauswirth, Manfred: *Software-Architekturen für Verteilte Systeme: Prinzipien, Bausteine und Standardarchitekturen für moderne Software (Xpert.press) (German Edition)*. 1. Springer, 7 2003. – URL http://amazon.com/o/ASIN/3540430881/. – ISBN 9783540430889

[Ellenberg 2011] Ellenberg, Jens: *Ontologiebasierte Aktivitätserkennung im Smart Home Kontext*, Hochschule für Angewandte Wissenschaften Hamburg, Masterthesis, 2011

[Fildebrandt 2012] Fildebrandt, Ulf: *Software modular bauen - Architektur von langlebigen Softwaresystemen - Grundlagen und Anwendung mit OSGi und Java.* 1. Auflage. Heidelberg : Dpunkt.Verlag GmbH, 2012. – ISBN 978-3-864-90019-8

[Friedewald u. a. 2009] Friedewald, Michael ; Raabe, Oliver ; Koch, Daniel J. ; Georgieff, Peter ; Neuhäusler, Peter: Zukunftsreport - Ubiquitäares Computing / Büro für Technologiefolgen-Abschätzungen beim Deutschen Bundestag. URL http://www.tab-beim-bundestag.de/de/pdf/publikationen/berichte/TAB-Arbeitsbericht-ab131.pdf, Mai 2009. – Forschungsbericht. (abgerufen am: 26.08.2013)

[GIRA] GIRA: *Intelligente Haustechnik.* – URL http://www.gira.de/gebaeudetechnik/systeme/knx-eib_system/knx-produkte/server/homeserver/zentral-steuern.html/. – (abgerufen am: 01.10.2013)

[Hardenack 2011] Hardenack, Frank: *Das intelligente Bett - Sensorbasierte Detektion von Schlafphasen*, Hochschule für Angewandte Wissenschaften Hamburg, Masterthesis, 2011

[Harrison u. a. 1976] Harrison, Michael A. ; Ruzzo, Walter L. ; Ullman, Jeffrey D.: Protection in operating systems. In: *Commun. ACM* 19 (1976), August, Nr. 8, S. 461–471. – URL http://doi.acm.org/10.1145/360303.360333. – ISSN 0001-0782

[Hull u. a. 1997] Hull, R. ; Neaves, P. ; Bedford-Roberts, J.: Towards situated computing. In: *Wearable Computers, 1997. Digest of Papers., First International Symposium on*, 1997, S. 146–153

[IANA] IANA: *Internet Assigned Numbers Authority - Internetpräzenz.* – URL http://www.iana.org/. – (abgerufen am: 29.08.2013)

[J. Sermersheim und Novell 2006] J. Sermersheim, Ed. ; Novell, Inc: Lightweight Directory Access Protocol (LDAP): The Protocol / Network Working Group. URL https://tools.ietf.org/html/rfc4511, June 2006. – RFC 4511. (abgerufen am: 18.06.2013)

[Joshua u. a. 1998] Joshua, David F. ; Franklin, David ; Flachsbart, Joshua: All gadget and no representation makes Jack a dull environment. In: *In AAAI Spring Symposium on Intelligent Environments. AAAI TR*, 1998, S. 155–160

[K. Zeilenga und Foundation 2006] K. Zeilenga, Ed. ; Foundation, OpenLDAP: Lightweight Directory Access Protocol (LDAP): Technical Specification Road Map / Network Working Group. URL https://tools.ietf.org/html/rfc4510, June 2006. – RFC 4510. (abgerufen am: 18.06.2013)

[Karstaedt 2012] Karstaedt, Bastian: *Kontextinterpretation in Smart Homes auf Basis semantischer 3D Gebäudemodelle*, Hochschule für Angewandte Wissenschaften Hamburg, Masterthesis, 2012

[Koubachi AG] Koubachi AG: *Koubachi - Online Store*. – URL http://store.koubachi.com/. – (abgerufen am 18.06.2013)

[Krumm 2009] Krumm, John (Hrsg.): *Ubiquitous Computing Fundamentals*. Chapman and Hall/CRC, 9 2009. – URL http://amazon.com/o/ASIN/1420093606/. – ISBN 9781420093605

[Kühn 2012] Kühn, Philipp: Ein interaktiver Couchtisch. In: *Projektbericht* (2012). – URL http://users.informatik.haw-hamburg.de/~ubicomp/projekte/master12-13-seminar/kuehn/bericht.pdf. – (abgerufen am: 19.06.2013)

[von Luck u. a. 2010] Luck, Prof. Dr. K. von ; Klemke, Prof. Dr. G. ; Gregor, Sebastian ; Rahimi, Mohammad A. ; Vogt, Matthias: A place for concepts of IT based modern living / University of Applied Sciences Hamburg. URL http://livingplace.informatik.haw-hamburg.de/content/LivingPlaceHamburg_en.pdf, 2010. – Forschungsbericht. (abgerufen am: 19.11.2012)

[Luckham 2002] Luckham, David: *The Power of Events - An Introduction to Complex Event Processing in Distributed Enterprise Systems*. ADDISON WESLEY Publishing Company Incorporated, 2002. – ISBN 978-0-201-72789-0

[Microsoft] Microsoft: *Microsoft Pixelsense - Surface 2*. – URL http://www.microsoft.com/en-us/pixelsense/whatsnew.aspx. – (abgerufen am: 19.06.2013)

[Mills u. a. 2010] Mills, D. ; Delaware, U. ; J. Martin, Ed. ; ISC ; Burbank, J. ; Kasch, W.: Network Time Protocol Version 4: Protocol and Algorithms Specification. URL http://tools.ietf.org/html/rfc5905, 2010. – RFC 9505. (abgerufen am: 17.06.2013)

[Mozer 1998] Mozer, Michael C.: The Neural Network House: An Environment that Adapts to its Inhabitants Department of Computer Science and Institute of Cognitive Science University of Colorado (Veranst.), URL http://www.cs.colorado.edu/~mozer/

Research/Selected%20Publications/reprints/nnhadapt.pdf, 1998, S. 110 – 114. – (abgerufen am: 20.06.2013)

[Najem 2013] Najem, Hosnia: *Kamerabasierte Suche von Objekten in einer Smart Home Umgebung*, Hochschule für Angewandte Wissenschaften Hamburg, Masterthesis, 2013

[Nginx] Nginx: *Nginx - Community*. – URL http://wiki.nginx.org/Main. – (abgerufen am: 06.09.2013)

[Norman 2004] Norman, Donald A.: Introduction to This Special Section on Beauty, Goodness, and Usability. In: *Human–Computer Interaction* 19 (2004), Nr. 4, S. 311–318. – URL http://www.tandfonline.com/doi/abs/10.1207/s15327051hci1904_1

[OSGi Alliance] OSGi Alliance: *OSGi - Alliance - Internetpräzenz*. – URL http://www.osgi.org/Main/HomePage. – (abgerufen am: 26.06.2013)

[OSGi Alliance 2011] OSGi Alliance: *OSGi Service Platform Core Specification*. April 2011. – URL http://www.osgi.org/download/r4v43/osgi.core-4.3.0.pdf. – (abgerufen am: 10.07.2013)

[Otto 2013] Otto, Kjell: *Aktuelle Entwicklungskonzepte zur Projektintegration in einem Smart Home anhand von Maven, OSGi und Drools Fusion*, Hochschule für Angewandte Wissenschaften Hamburg, Masterthesis, 2013

[Otto und Voskuhl 2010] Otto, Kjell ; Voskuhl, Sören: Entwicklung einer Architektur für den Living Place Hamburg. In: *Projektbericht* Sommersemester 10 (2010), S. 21. – URL http://users.informatik.haw-hamburg.de/~ubicomp/projekte/master2010-proj1/otto_voskuhl.pdf. – (abgerufen am: 25.11.2012)

[Papazoglou u. a. 2008] Papazoglou, Michael P. ; Traverso, Paolo ; Dustdar, Schahram ; Leymann, Frank: *SERVICE-ORIENTED COMPUTING: A Research Roadmap*. 2008

[Papazoglou 2003] Papazoglou, M.P.: Service-oriented computing: concepts, characteristics and directions. In: *Web Information Systems Engineering, 2003. WISE 2003. Proceedings of the Fourth International Conference on*, 2003, S. 3–12

[Pautz 2011] Pautz, Alexander: *Kabellose, stromsparende Sensornetzwerke im Bereich Ambient Intelligence*, Hochschule für Angewandte Wissenschaften Hamburg, Masterthesis, 2011

[Philips hue] Philips hue: *Philips hue*. – URL https://www.meethue.com/de-DE. – (abgerufen am: 01.10.2013)

[Play] Play: *Play Framework*. – URL http://www.playframework.com/. – (abgerufen am: 16.08.2013)

[QIVICON] QIVICON: *Smart Home für alle*. – URL https://www.qivicon.com/start/. – (abgerufen am: 01.10.2013)

[Quast 2011] Quast, Oliver: *Berührungslose Schlafphasenerkennung zur Integration in ein Smart-Home*, Hochschule für Angewandte Wissenschaften Hamburg, Bachelorthesis, 2011

[ROCKETHOME] ROCKETHOME: *Smart-Home-Ecosystem, die Endkunden begeistert*. – URL http://www.rockethome.de/?locale=de_DE. – (abgerufen am: 01.10.2013)

[Roy u. a. 2007] Roy, Abhishek ; Das, S.K. ; Basu, K.: A Predictive Framework for Location-Aware Resource Management in Smart Homes. In: *Mobile Computing, IEEE Transactions on* 6 (2007), Nr. 11, S. 1270–1283. – ISSN 1536-1233

[Sandhu u. a. 1996] Sandhu, Ravi S. ; Coyne, Edward J. ; Feinstein, Hal L. ; Youman, Charles E.: Role-Based Access Control Models. In: *Computer* 29 (1996), Februar, Nr. 2, S. 38–47. – URL http://dx.doi.org/10.1109/2.485845. – ISSN 0018-9162

[Sandhu 1993] Sandhu, R.S.: Lattice-based access control models. In: *Computer* 26 (1993), nov., Nr. 11, S. 9 –19. – ISSN 0018-9162

[Schilit u. a. 1994] Schilit, B. ; Adams, N. ; Want, R.: Context-aware computing applications. In: *Mobile Computing Systems and Applications, 1994. Proceedings., Workshop on*, dec 1994, S. 85 –90

[Schilit und Theimer 1994] Schilit, B.N. ; Theimer, M.M.: Disseminating active map information to mobile hosts. In: *Network, IEEE* 8 (1994), Nr. 5, S. 22–32. – ISSN 0890-8044

[Seamless Interaction] Seamless Interaction: *Seamless Interaction - Projekte*. – URL http://www.seamlessinteraction.com/projects.html. – (abgerufen am: 19.06.2013)

[SmartHome Initiative Deutschland] SmartHome Initiative Deutschland: *SmartHome Initiative Deutschland - Internetpräzenz*. – URL http://www.smarthome-deutschland.de/index. – (abgerufen am: 20.06.2013)

[Snyder u. a. 2011] Snyder, B. ; Bosnanac, D. ; Davies, R.: *ActiveMQ in Action*. Manning Publications Company, 2011 (In Action Series). – URL http://books.google.de/books?id=_jjCPwAACAAJ. – ISBN 9781933988948

[SpringSource] SpringSource: *Spring Projects: SPRING LDAP*. – URL http://www.springsource.org/ldap. – (abgerufen am: 09.07.2013)

[Steggles und Gschwind] Steggles, Pete ; Gschwind, Stephan: THE UBISENSE SMART SPACE PLATFORM. . – URL http://www.pervasive.ifi.lmu.de/adjunct-proceedings/demo/p073-076.pdf. – (abgerufen am: 17.06.2013)

[Technische Universität Darmstadt] Technische Universität Darmstadt: *PRORETA 3*. – URL http://www.proreta.tu-darmstadt.de/proreta_1/startseite_proreta/index.de.jsp. – (abgerufen am: 23.09.2013)

[Teske 2011] Teske, Philipp: *Ein Multisensor-System zur Sturzerkennung*, Hochschule für Angewandte Wissenschaften Hamburg, Masterthesis, 2011

[The Apache Software Foundation] The Apache Software Foundation: *Apache Directory Project*. – URL http://directory.apache.org/apacheds/. – (abgerufen am: 26.06.2013)

[The JBoss Drools team 2012] The JBoss Drools team: *Drools Fusion User Guide*. November 2012. – URL http://docs.jboss.org/drools/release/5.5.0.Final/drools-fusion-docs/pdf/drools-fusion-docs.pdf. – (abgerufen am: 14.08.2013)

[Tsolkas und Schmidt 2010] Tsolkas, Alexander ; Schmidt, Klaus: *Rollen und Berechtigungskonzepte - Ansätze für das Identity- und Access Management im Unternehmen*. 2010. Aufl. Wiesbaden : Vieweg+Teubner Verlag, 2010. – ISBN 978-3-834-81243-8

[Ubisense] Ubisense: *Ubisense - Internetpräzenz*. – URL http://www.ubisense.net/en/. – (abgerufen am: 17.06.2013)

[Voskuhl 2011] Voskuhl, Sören: *Modellunabhängige Kontextinterpretation in einer Smart Home Umgebung*, Hochschule für Angewandte Wissenschaften Hamburg, Masterthesis, 2011

[Walzer u. a. 2007] Walzer, Karen ; Schill, Alexander ; Löser, Alexander: Temporal constraints for rule-based event processing. In: *Proceedings of the ACM first Ph.D. workshop in CIKM*. New York, NY, USA : ACM, 2007 (PIKM '07), S. 93–100. – URL http://doi.acm.org/10.1145/1316874.1316890. – ISBN 978-1-59593-832-9

[Weber u. a. 2010] Weber, Bernd ; Baumgartner, Patrick ; Braun, Oliver: *OSGi für Praktiker - Prinzipien, Werkzeuge und praktische Anleitungen auf dem Weg zur "kleinen SOA"*. München : Hanser Fachbuchverlag, 2010. – ISBN 978-3-446-42094-6

[Weiser 1991] Weiser, Mark: The Computer for the 21st Century. In: *Scientific American* 265 (1991), September, Nr. 3, S. 94f. (Intl. ed. 66–75). – ISSN 0036-8733 (print), 1946-7087 (electronic)

[Zörner 2008] Zörner, Stefan: *LDAP für Java-Entwickler - Einstieg und Integration.* 3. Aufl. Unterhaching : Entwickler.press, 2008. – ISBN 978-3-939-08407-5

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, 02.12.2013 Sven Boris Bornemann