Fakultät für Informatik

Bachelorarbeit in Wirtschaftsinformatik

Konzeption und Prototypimplementierung von Licht- und Audio-Komponenten für eine Internet-of-Things Mikro-Informationsstrahler-Umgebung auf der Basis von RaspberryPi-Mini-Rechnern

Vitus Lehner

Fakultät für Informatik

Bachelorarbeit in Wirtschaftsinformatik

Konzeption und Prototypimplementierung von Licht- und Audio-Komponenten für eine Internet-of-Things Mikro-Informationsstrahler-Umgebung auf der Basis von RaspberryPi-Mini-Rechnern

Concept and Implementation of Light and Audio Components for an Internet-of-Things Information Radiator Environment based on Raspberry Pi Mini Computers

Bearbeiter:	Vitus Lehner
Aufgabensteller:	PD Dr. Georg Groh
Betreuer:	Prof. Dr. Michael Koch
Abgabedatum:	15.09.2017

Ich versichere, dass ich diese Bachelorarbeit selbstständig verfasst und nur die angegebenen Quellen und Hilfsmittel verwendet habe.

_____

Datum, Unterschrift

Zusammenfassung

Im Rahmen des Projekts UrbanLife+ wird untersucht, wie sich die Teilhabe älterer Menschen im öffentlichen, städtischen Raum erhöhen lässt und wie Seniorinnen und Senioren zu Aktivitäten angeregt werden können. Teil des Projekts sind bspw. interaktive Wandbildschirme (CommunityMirrors), die Informationen zu interessanten Orten, Dienstleistungen oder Veranstaltungen zeigen. Durch Ankopplung persönlicher Endgeräte, wie Smartphones, werden die Bildschirminhalte individualisiert und an die Nutzerbedürfnisse angepasst, so können sich Senioren über geeignete Aktivitäten informieren und sich neuen Herausforderungen stellen. Damit Nutzern auch auf Wegen von und zu Orten das Gefühl von Sicherheit und technologischer Unterstützung vermittelt wird, sollen städtebauliche Objekte, wie Straßenlaternen, Parkbänke und Bushaltestellen, ‚smartifiziert' werden. In dieser Arbeit werden ein Konzept und ein Prototyp über eine Klasse solcher ‚Smart Urban Objects' vorgestellt, die hier Mikroinformationsstrahler genannt wird. Mikroinformationsstrahler sind lediglich mit einfachen Licht- und Audio-Komponenten ausgestattet und sollen Nutzer via iBeacons erkennen, um sie adäquat zu unterstützen. Durch die eingeschränkten Benutzerschnittstellen ergeben sich besondere Herausforderungen bzgl. der Interaktion mit solchen Strahlern. Diese Arbeit zeigt Anforderungen und Möglichkeiten der Interaktion auf und stellt ein Architekturkonzept für das Szenario in einer Internet-of-Things-artigen Umgebung vor. Der außerdem entwickelte Prototyp illustriert die Umsetzung des Konzepts anhand eines Raspberry Pi Minicomputers und setzt dabei auf aktuelle Technologien, wie MQTT, synthetisierte Sprachausgaben und iBeacons.

Inhaltsverzeichnis

Abbildungsverzeichnis

Abkürzungsverzeichnis

CC	Cloud Computing
EDA	Event-Driven Architecture
EV	Environment Variable (Umgebungsvariable eines Betriebssystems)
FC	Fog Computing
HTTP	Hypertext Transfer Protocol
IoT	Internet of Things
IP	Internet Protocol
IS	Informationsstrahler (Mikro, Mini und Makro, also auch digitale, interaktive Informationstafeln)
JSON	JavaScript Object Notation
MCI	Mensch-Computer-Interaktion
MIR	Micro Information Radiator (englisches Äquivalent für MIS)
MIS	Mikroinformationsstrahler
MQTT	Message Queue Telemetry Transport
ÖPNV	Öffentlicher Personennahverkehr
REST	Representational State Transfer
RFID	Radio Frequency Identification
RPi	Raspberry Pi
SOA	Serviceorientierte Architektur
TCP	Transmission Control Protocol
UbiComp	Ubiquitous Computing
XML	Extensible Markup Language

1. Einleitung

1.1. Motivation

Das Projekt UrbanLife+ richtet sich an ältere Menschen und verfolgt das Ziel, Seniorinnen und Senioren im öffentlichen, städtischen Raum bei der Bewältigung ihres Alltags zu unterstützen und zu Aktivitäten auch außerhalb ihrer Komfortzone anzuregen. Teil des Projekts sind bspw. große Informationsdisplays, die Informationen zu interessanten Orten, Dienstleistungen oder Veranstaltungen zeigen und mit welchen interagiert werden kann. In Verbindung mit persönlichen Endgeräten (Smartphone, Smartcard, …) und Konnektivität zum Infodisplay (Bluetooth, iBeacon, Wifi, …) können die Inhalte individualisiert und an die Bedürfnisse und Ziele des Nutzers angepasst werden. Senioren/innen können bspw. über eine nahegelegene Veranstaltung informiert werden und sich diese als Ziel festlegen. Um die Nutzer auch auf dem Weg zu oder von einem Ort unterstützen zu können, sollen auch städtebauliche Objekte, wie Straßenlaternen, Parkbänke, Ampeln oder Bushaltestellen, ‚smartifiziert' werden. Diese „Smart Urban Objects" können Mini-Infoscreens, aber auch reduzierte Informationsstrahler auf Basis von Licht und Ton sein, die ihrerseits den Nutzer erkennen und versuchen, diesen zu unterstützen. Bei diesen ergeben sich auch besondere Herausforderungen hinsichtlich der Art und Weise, wie mit diesen Objekten interagiert werden kann (Mensch-Computer-Interaktion).

1.2. Aufgabenstellung

Ziel dieser Arbeit ist es, Licht- und Audio-Komponenten für eine Internet-of-Things Mikro-Informationsstrahler-Umgebung zu konzipieren und prototypisch zu implementieren. Als technische Grundlage sollen dabei Raspberry Pi Mini-Computer zum Einsatz kommen. Für ein ausgewähltes Nutzungsszenario aus dem UrbanLife+ Projekt gilt es insbesondere herauszufinden, welche Möglichkeiten der Mensch-Computer-Interaktion sich für solche einfachen Informationsstrahler ergeben. Dabei sollen auch die Anforderungen für deren Anwendung und Einsatz ermittelt werden. Möglichkeiten und Anforderungen sollen auch bezüglich der Internet-of-Things-artigen Architektur geklärt werden. Dabei spielt insbesondere die Kommunikation zwischen Komponenten wie dem Service Backend und persönlichen Endgeräten eine Rolle.

1.3. Erwartetes Ergebnis

Diese Arbeit stellt ein Konzept über Möglichkeiten der Mensch-Computer-Interaktion mit einfachen Licht- und Audio-Komponenten vor. Die Möglichkeiten werden dabei in einem visionsartigen Szenario über UrbanLife+ dargelegt. Ausgehend von diesem Szenario wird ein technisches Architekturkonzept präsentiert, das dessen Anforderungen gerecht wird. Dabei wird sich an den Paradigmen, die sich im Internet of Things finden lassen, orientiert.

Auf dem Konzept aufbauend entsteht außerdem ein Raspberry Pi Prototyp, der Schlüsselfunktionen und -methoden aus dem Konzept demonstriert und als Diskussionsbasis für weitere Entwicklungen dient. Der Prototyp soll das Architekturkonzept anwenden und die Funktionsweise erläutern. Neben der Programmierung der Hardware-Komponenten gehört dazu insbesondere auch die Entwicklung einer Software, in der aktuelle Methoden und Technologien verwirklicht werden. Die Software soll andere Komponenten des UrbanLife+ Projekts anbinden (z.B. Backend, Smartphones, etc.) und mit ihnen kommunizieren. Anhand der prototypischen Implementierung und Tests wird außerdem kurz evaluiert, ob das Architekturkonzept passend für die Mikroinformationsstrahler ist.

1.4. Vorgehen zur Erreichung des Ziels

Zunächst werden verwandte, wissenschaftliche Arbeiten recherchiert, um einen Überblick über den Stand der Wissenschaft zu erhalten und Erkenntnisse aufzugreifen. Besonders interessant sind dabei Grundlagen der MCI und Benutzerschnittstellen einfacher Informationsstrahler, sowie das Internet-of-Things.

Anschließend wird eine Reihe von Anforderungen und Voraussetzungen formuliert, die sich an das zu erarbeitende Konzept richten oder die im Konzept als gegeben betrachtet werden können. Dabei werden sowohl Kontext, Szenario und Interaktion, als auch technische und architektonische Kriterien aufgegriffen.

Auf Basis des Stands der Wissenschaft und der Anforderungen wird ein Konzept über Mikroinformationsstrahler beschrieben. Dabei werden zunächst Interaktionsmöglichkeiten mit einfachen Licht- und Audio-Komponenten abgeleitet und diese in ein erweitertes Szenario mit Mikroinformationsstrahlern eingebaut. Davon ausgehend wird ein Architekturkonzept entwickelt, das sich auf fachlicher und technischer Ebene mit der Thematik auseinandersetzt und aktuelle technologische Möglichkeiten aus dem Bereich des Internet of Things aufgreift.

Das Konzept ist die Grundlage für eine prototypische Implementierung. Wie ein Raspberry Pi Prototyp entwickelt wurde, wird in im vorletzten Kapitel beschrieben. Darin wird die Funktionsweise der Software genauer erklärt und die Grenzen des Prototyps bzw. der Architektur ausgemacht.

Zuletzt werden die Ergebnisse dieser Arbeit reflektiert und ein Ausblick auf künftige Entwicklungen gewagt.

2. Stand der Wissenschaft

Der Stand der Wissenschaft gibt einen Überblick über aktuelle Forschungen und Entwicklungen, die für diese Arbeit relevant sind oder auf welchen diese Arbeit fußt. Hierfür werden zunächst grundlegende Trends der Interaktion zwischen Mensch und Computer dargelegt. Anschließend werden konkrete Fallbeispiele und Konzepte der Mensch-Computer-Interaktion im öffentlich-urbanen Raum gezeigt. Darüber hinaus werden Grundlagen des Internet of Things vorgestellt, die insbesondere die technische Basis für das Konzept liefern.

Dieses Kapitel ist unterteilt in drei Abschnitte. Bei der Recherche wurden unterschiedliche Methoden angewandt. In den Abschnitten 2.1 und 2.3 werden Grundlagen der Mensch-Computer-Interaktion bzw. des Internet of Things gezeigt. Hier wurde hauptsächlich Grundlagenliteratur auf den entsprechenden Gebieten recherchiert. Suchbegriffe waren neben „human computer interaction“ und „internet of things“ etwa: iot architecture, ubiquitous computing, cloud computing, software architecture, fog computing, event driven architecture, service oriented architecture und iot technology.

Bei der Grundlagenliteratur wurde sich durchaus an Zitationszahlen orientiert. Vergleichsweise hohe Zitierquoten wurden als vertrauenswürdiger eingestuft. Auch die in Veröffentlichungen angegebene Literatur wurde in Betracht gezogen. Die folgenden Datenbanken und Kataloge wurden bei der Recherche genutzt: ACM, Google Scholar, Google, Universitätsbibliothek Technische Universität München, Science Direct/Elsevier, Springer Link und IEEE Xplore.

Abschnitt 2.2 zeigt hauptsächlich Klassen von Interaktionsmöglichkeiten im urbanen Raum. Hier wurde verstärkt versucht einen Überblick über die Vielfalt der Interaktionen zu erhalten. Dazu wurde mit diversen Suchbegriffen und -kombinationen gearbeitet: smart urban objects, interactive city, smart city, smart street furniture, speaking street furniture, urban hci, environment hci, urban assistance und Anderen. Hier wurde weniger auf Zitationszahlen geachtet, da dies für einen Überblick über existierende Arbeiten vom Autor für weniger relevant befunden wurde.

2.1. Grundlagen der Mensch-Computer-Interaktion

Dieser Abschnitt behandelt einige Grundlagen der Mensch-Computer-Interaktion (kurz MCI). Dabei wird der Gegenstand dieser Arbeit zunächst in den Kontext einer größeren, technologischen MCI-Vision gesetzt. Anschließend werden Klassen von Benutzerschnittstellen und Konzepte der MCI erläutert.

2.1.1. Ubiquitous Computing

Insbesondere Hardwareverbesserungen und -neuentwicklungen ermöglichen hohe Übertragungsraten in kabelgebundenen und kabellosen Netzwerken, sowie immer kleinere, kostengünstigere Geräte, die in unsere physische Umgebung integriert sind. In einer solchen Umgebung scheint Informations- und Kommunikationstechnologie eine allgegenwärtige Rolle zu spielen, so ist „Ubiquitäres Computing“ (engl. Ubiquitous Computing, kurz UbiComp) mittlerweile eine etablierte Bezeichnung (Poslad, 2009, pp. 1–2).

Mark Weiser verwendet den Begriff erstmals 1991, indem er eine Welt beschreibt, in der der Mensch in einer nahtlos von einer Vielzahl von Computern gestützten Umgebung lebt und arbeitet (Poslad, 2009, pp. 8–9; Weiser, 1999). Die Verteilung und Vernetzung vieler kleiner Computer bedeutet zunächst eine höhere Komplexität des Systems, so spielen neuartige

Interaktionskonzepte auch in Weisers Vision eine elementare Rolle. Stefan Poslad bemerkt, dass konventionelle Benutzerschnittstellen aus bspw. Bildschirm, Tastatur und Maus für diese Vision zu aufdringlich und einnehmend sind und leitet daher unter anderem die Anforderung ab, dass Mensch-Computer-Interaktion für UbiComp versteckter (Poslad, 2009, pp. 8–9) oder nach Weiser gar unsichtbar (Weiser, 1994) werden muss.

Darin lässt sich auch ein Paradigmenwechsel von expliziter MCI, d.h. kontextunbewussten Computern und vom Menschen initiierte Aktionen, hin zu impliziter Interaktion, d.h. kontextbewussten Computern und von Computern initiierte Aktionen, erkennen. Interaktion versucht man dabei auch durch dem Menschen natürlich wirkende Schnittstellen, wie Gesten oder Sprache, zu implementieren (Poslad, 2009, pp. 136–138).

Das hier erarbeitete Konzept zu Mikroinformationsstrahlern stößt in Richtung des Felds des UbiComp. Aus diesem Grund folgen die nachfolgend vorgestellten Benutzerschnittstellen und Interaktionsdesigns überwiegend Gesichtspunkten impliziter und natürlicher Interaktion. Sie erreichen dabei noch nicht zwangsläufig, wie in Weisers Vision angedacht, den Grad der Unsichtbarkeit (Weiser, 1994). Da UbiComp allerdings der Post-WIMP-Ära zuzuordnen ist (Jacob et al., 2008; van Dam, 1997), wird hier nicht näher auf konventionelle grafische Benutzerschnittstellen mit Tastatur und Maus oder Ähnliches eingegangen. Abschließend werden kurz weitere aufkommende Konzepte der Interaktion dargelegt.

2.1.2. Benutzerschnittstellen

Jonathan Grudin, Mitglied der ACM SIGCHI Academy, identifiziert Hardwarekapazitäten als wesentlichen Treiber für das Aufkommen neuer Formen der Interaktion (Grudin, 2012, pp. liv–lv). Aus diesem Grund liegt der Fokus dieses Abschnitts auf jüngeren Benutzerschnittstellen, die erst in den letzten 10-15 Jahren gesellschaftlich-technologische Durchbrüche erzielten, oder die sich noch in der Forschung befinden.

Die hier vorgestellten Klassen und Beispiele von Benutzerschnittstellen basieren auf einer Zusammenstellung von Poslad (Poslad, 2009, pp. 143–157). Die folgende Liste versucht keinen Anspruch auf Abgeschlossenheit zu erheben, vielmehr zeigt sie einige grundverschiedene Ein- und Ausgabemodalitäten und wie sie bereits in Verwendung sind oder sein können. Sie schafft einen Überblick über heutige Möglichkeiten der Interaktion, zugleich klärt sie erste technologische oder materielle Anforderungen.

2.1.2.1. Gestenschnittstellen

Gestenschnittstellen erlauben dem Benutzer Interaktionen auf Basis natürlicher Gesten mit Fingern, Händen, Armen und anderen Körperteilen. Mit Gesten können Informationen kommuniziert (z.B. Daumen hoch), Objekte manipuliert oder Informationen eingeholt werden. Gesten können technisch bspw. durch anziehbare Geräte wie Handschuhe oder Anzüge, Kameras und Bilderkennung oder Sensoren am Körper erkannt werden. Dabei können auch Touchscreen-Eingaben als (2D-) Gesten betrachtet werden. Eine 3D-Geste wäre entsprechend z.B. Winken oder ein Händedruck (Hinckley & Wigdor, 2012, p. 117).

Die Einsatzzwecke von Gestenschnittstellen sind vielfältig und reichen von Zeichenspracheerkennung von Hörbeeinträchtigten über die grundsätzliche Steuerung von Computern bis hin zu Spielkonsolenanwendungen (Poslad, 2009, p. 145). Kommerziell

erhältliche Produkte sind etwa Microsoft Kinect für Xbox One[1], Intel RealSense SR300[2], Leap Motion Controller[3] und Myo Gesture Control Armband[4].

2.1.2.2. Touchscreens

Touchscreens gehören einer Kategorie von Benutzerschnittstellen an, welche Ein- und Ausgabegeräte in einer einzigen Schnittstelle kombiniert. Durch Berührung des Bildschirms mit bspw. einem Stift oder Fingern können Aktionen auf virtuellen Objekten ausgelöst werden. Der Benutzer erhält auf selbigem Bildschirm Feedback über die von ihm getätigten Aktionen. Es gibt u.a. resistive und kapazitive Touchscreen-Technologien (Poslad, 2009, p. 149).

Touchscreens sind heutzutage weit verbreitet und kommen bspw. in (Geld-)Automaten, Kfz-Navigationssystemen, Spielkonsolen und Mobiltelefonen zum Einsatz.

2.1.2.3. Gegenständliche Benutzerschnittstellen

Gegenständliche Benutzerschnittstellen (engl. Tangible User Interface, kurz TUI) gewannen mit dem Projekt und der Vision „Tangible Bits" unter Leitung von Hiroshi Ishii an Bedeutung (Ishii & Ullmer, 1997; Poslad, 2009, p. 149). Dabei werden Sensoren und Auslöser an physische Objekte gekoppelt. Manipulationen an den Objekten können dann als Eingabe interpretiert werden. Bspw. kann die Position eines mit RFID-Tag markierten, bewegten Objekts innerhalb eines Raumes bestimmt und die daraus gewonnenen Informationen dargestellt werden. Wie bei Touchscreens sind auch hier Fusionen aus Ein- und Ausgabegerät möglich. Ein Beispiel ist die Computermaus, die Verschiebungen des Mauszeigers wiederum selbst verkörpert.

Häufig werden TUIs durch visuelle Projektionen unterstützt. Das Projekt metaDESK besteht aus einer transparenten (Tisch-)Oberfläche, auf welche von unten visuell Inhalte projiziert werden. Darauf können physische Objekte/Icons als Instrumente benutzt werden, die durch Sensoren erfasst werden (Ishii & Ullmer, 1997, pp. 237–238).

2.1.2.4. Auditive Benutzerschnittstellen

Auditive Schnittstellen basieren auf Lautsprechern und Mikrofonen und können vielfältige Möglichkeiten der Interaktion eröffnen. Man kann zwischen Sprach- und Nicht-Sprach-Schnittstellen unterscheiden. Letztere sind bspw. Musik, Umgebungsgeräusche oder Soundeffekte. Nicht-Sprach-Ausgaben können in multimodalen Anwendungen eine mächtige Feedbackmöglichkeit bieten (Hoggan & Brewster, 2012, pp. 211–212). Besonders relevant ist diese Schnittstelle auch für Sehgeschädigte, die über Sounds Informationen aufnehmen können.

Sprachbasierte Schnittstellen hingegen versuchen einen zwischenmenschlichen Dialog zu emulieren, d.h. Mensch und Computer sprechen miteinander (Karat, Lai, Stewart & Yankelovich, 2012, pp. 367–368). Zur Sprachausgabe kann synthetisierte oder aufgenommene Sprache zum Einsatz kommen. Bei der Spracheingabe stellen unterschiedliche Aussprachen und die Genauigkeit der Spracherkennung nur zwei der größeren, technischen Herausforderungen dar, die es bei der Implementierung zu bewältigen gilt. In einfachen Varianten spricht der Benutzer

---

[1] http://www.xbox.com/de-DE/xbox-one/accessories/kinect (letzter Abruf: 31.05.2017)

[2] https://click.intel.com/intelrealsense-developer-kit-featuring-sr300.html (letzter Abruf: 31.05.2017)

[3] https://store-eur.leapmotion.com/products/leap-motion-controller (letzter Abruf: 31.05.2017)

[4] https://www.myo.com/ (letzter Abruf: 31.05.2017)

dabei lediglich einzelne (Befehls-)Worte aus, in komplexeren sind es auch Sätze (siehe natürliche Sprachschnittstellen im nächsten Abschnitt). Sprachschnittstellen werden bspw. bei automatischen Telefon-Anrufbeantwortern und zunehmend auch in Smartphones oder Navigationssystemen eingesetzt. Text-to-Speech-Systeme ermöglichen Sehgeschädigten den barrierefreien Zugang zu digitalen Inhalten, wie Webseiten. Kommerziell erhältliche Produkte sind bspw. Siri aus Apples iOS[5] oder Amazon Echo mit dem Sprachassistenten Alexa[6].

2.1.2.5. Natürliche Sprachschnittstellen

Expressive Sprachen mit wohldefinierter Syntax und Grammatik ermöglichen eine einfache Erkennung und Verarbeitung von Interaktionen, erfordern aber, dass der Benutzer die Sprache beherrscht bzw. erlernt (Poslad, 2009, pp. 151–152). Bei natürlichen Sprachschnittstellen (engl. Natural Language Interface, kurz NLI) wird versucht diese Barriere zu umgehen, indem Benutzer eine Sprache verwenden, der sie bereits mächtig sind - etwa die Muttersprache. Der Benutzer kann dabei auch ganze Sätze verwenden, um sich auszudrücken. Zu bemerken ist, dass sich NLI nicht auf auditive Kommunikation beschränkt, sondern auch auf Text- oder Handschriftbasis möglich ist. Dabei ergeben sich neben den Problemen auditiver Schnittstellen auch: Sätze können mehrdeutige Bedeutungen haben, Ausgaben können nur relativ einfache, wenig komplexe Informationen enthalten und im Kontext von Audioausgaben sind Informationen nur flüchtig und kurzzeitig erfassbar (Karat et al., 2012, pp. 370–373; Poslad, 2009, pp. 151–152).

2.1.2.6. Gehirn-Computer-Schnittstellen

Die Forschung über Gehirn-Computer-Schnittstellen (engl. Brain-Computer-Interface, kurz BCI) beschäftigt sich damit, wie das menschliche Nervensystem angebunden werden kann, um Gedanken, Emotionen und Gefühle direkt als Eingabe für Computer zu nutzen und diese somit über Gedanken zu steuern (Poslad, 2009, pp. 155–157). Die Schnittstelle kann dabei am Körper getragen oder implantiert werden, Orte können das Gehirn, aber auch andere Körperteile sein. Anwendung könnte die Technologie bspw. bei der Steuerung von Robotern finden.

2.1.2.7. Augmented und Virtual Reality

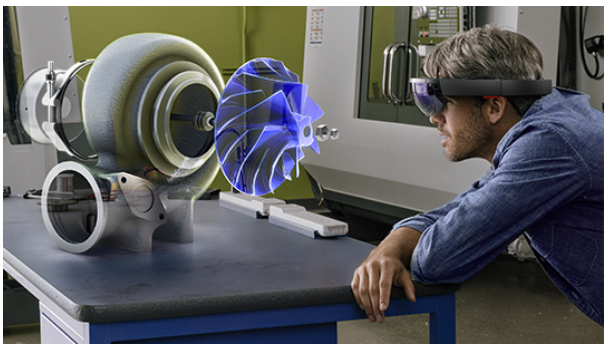

Abbildung 1 - Microsoft Hololens Szenario

Die erweiterte Realität (engl. Augmented Reality, kurz AR) befasst sich mit der Erweiterung der realen Welt durch virtuelle Komponenten, die auf digitale Weise reale Objekte aktivieren. Dabei werden Bilder über die tatsächliche Umgebung projiziert. Virtuelle und reale Welten werden vereint. Abbildung 1 zeigt ein Anwendungsszenario der Entwicklung einer mechanischen Komponente[7]. Die virtuelle Realität (engl.

[5] https://www.apple.com/de/ios/siri/ (letzter Abruf: 31.05.2017)

[6] https://www.amazon.de/Amazon-SK705DI-Echo-Schwarz/dp/B01GAGVCUY (letzter Abruf: 31.05.2017)

[7] Quelle: https://compass-ssl.surface.com/assets/22/33/22331909-9945-4a15-9e34-a7ba1a280aa5.jpg?n=Overview_Hero_539_img_new.jpg (letzter Abruf: 14.08.2017)

Virtual Reality, kurz VR) bildet einen Spezialfall der AR, bei welchem die reale Welt gänzlich ausgeblendet und der Mensch in eine reine Computersimulation versetzt wird (Poslad, 2009, pp. 152–153).

Häufig werden VR und AR durch auf den Kopf gesetzte Anzeigesysteme oder Brillen realisiert und durch Handgeräte ergänzt. Kommerzielle Produkte sind etwa Oculus Rift[8] und Microsoft Hololens[9]. Ihr Einsatz ist sowohl in Spielekonsolenanwendungen, als auch im produzierenden Gewerbe oder dem medizinischen Sektor denkbar.

2.1.3. Multimodalität

Systeme haben multimodale Schnittstellen, wenn sie zwei oder mehr Eingabetypen, bspw. Sprache und Berührung, kombiniert vom Benutzer entgegennehmen können und multimedial antworten (Oviatt, 2012). So wie Menschen in einem Gespräch mehrere Informationsquellen gleichzeitig wahrnehmen, etwa Sprache und Gesten, können multimodale Anwendungen eine natürlichere Interaktion ermöglichen (Zühlke, 2012, pp. 227–229). Dadurch ergeben sich u.a. mehr Flexibilität bei der Art und Weise, wie mit Computern gearbeitet wird, und bessere Möglichkeiten, um komplexer werdende und multifunktionale Systeme, zu bedienen.

2.1.4. Kriterien & Design

In der Literatur finden sich zahlreiche Konzepte, um die Benutzbarkeit von MCI-Designs zu evaluieren.

Nach ISO-Norm 9241-11 ist Benutzbarkeit definiert als das Ausmaß in dem ein Produkt dazu genutzt werden kann, dass spezifische Nutzer spezifische Ziele mit Effektivität, Effizienz und Zufriedenheit erreichen (Poslad, 2009, p. 161). Effektivität kann dabei als Grad verstanden werden, zu dem die Ziele erreicht werden. Effizienz kann bspw. mit der Zeit gemessen werden, die benötigt wird, um eine Aufgabe zu erfüllen.

Diese Arbeit befasst sich hauptsächlich mit der Konzeption eines MCI-Designs, eine nachträgliche Evaluation ist hier nicht vorgesehen. Aus diesem Grund liegt der Fokus an dieser Stelle auf Methoden, die sich schon während der Design-Phase anwenden lassen.

In wissenschaftlichen Arbeiten wird eine Vielzahl an Kriterien und Prinzipien zur Erreichung der MCI-Ziele vorgestellt. Dix et al. haben eine umfangreiche Sammlung an MCI-Design-Regeln zusammengestellt (Dix, Finlay, Abowd, & Beale, 2004, Chapter 7). In der Literatur als einfaches, knappes Regelset verbreitet sind die acht goldenen Regeln des Interaktionsdesigns von Shneiderman (Dix et al., 2004, Chapter 7.5.1).

In kurzen Stichpunkten zusammengefasst bestehen Shneidermans Regeln aus: dem Streben nach Konsistenz, dem Ermöglichen von Shortcuts für frequentierte Nutzungen, dem Anbieten von informativem Feedback, Dialogdesigns aus offenkundig beendbaren Sequenzen, einfacher Fehlervermeidung und Fehlerhandhabung, dem Erlauben von Aktionsrevokationen, der Bewahrung der Nutzerkontrolle über das System, sowie der Reduktion der Belastung des Kurzzeitgedächtnisses. Die Liste kann als Checkliste für Interaktionsdesigns verstanden werden,

[8] https://www.oculus.com/rift/ (letzter Abruf: 31.05.2017)

[9] https://www.microsoft.com/de-de/hololens (letzter Abruf: 31.05.2017)

erhebt jedoch keinen Anspruch daran, für jeden Anwendungsfall passend zu sein, und muss für jede Situation eigens interpretiert werden (Dix et al., 2004, Chapter 7.5.1).

1990 stellt eine Gruppe der North Carolina State University eine noch abstraktere Regelliste vor, die es auf Design im Generellen absieht. Die Prinzipien sind: gleichberechtigte Benutzbarkeit, flexible Benutzbarkeit, einfache und intuitive Bedienung, aufnahmefähige Informationen, Fehlertoleranz, geringer physischer Einsatz, sowie größen- und raumunabhängige Benutzbarkeit. Wie Shneidermans Regeln ist auch diese Zusammenstellung als Checkliste anwendbar, jedoch nicht für jede Situation passend (Dix et al., 2004, Chapter 10.2).

2.2. Mensch-Computer-Interaktion im urbanen Raum

Es gibt im urbanen Raum bereits einige Entwicklungen, die neuartige Konzepte der MCI aufgreifen. Bedeutend für diese Arbeit sind dabei insbesondere Konzepte über fest installierte, digitale, städtebauliche Objekte, mit welchen Interaktionen möglich sind. Solche Objekte können funktional durch persönliche Nutzerendgeräte (Smartphones, Wearables, ...) ergänzt werden.

Im Nachfolgenden werden einige Projekte und wissenschaftliche Arbeiten über interaktive Geräte im urbanen Raum vorgestellt. Sie stammen u.a. aus dem Bereich Smart Street Furniture. Die Arbeiten wurden nicht hauptsächlich auf Grund ihrer wissenschaftlichen Relevanz oder Zitationsrate ausgewählt, vielmehr beabsichtigt dieser Abschnitt eine gewisse Vielfalt an bestehender urbaner Interaktion aufzudecken und von der möglicherweise gewonnenen Rezeption der Projekte zu lernen. Dieser Abschnitt dient gewissermaßen als Inspirationsquelle für das Interaktionsdesign der Mikroinformationsstrahler. Neben wissenschaftlichen Projekten wurden daher auch teils private bzw. gewerbliche Entwicklungen aufgenommen.

2.2.1. Smart Street Furniture

Als Stadtmöbel (engl. Street Furniture) versteht man im Stadtgebiet aufgestellte kleinere bauliche Elemente, die funktionalistischen städteplanerischen Vorstellungen entsprechen und dem Stadtbild ein bestimmtes Gepräge geben sollen (“Duden | Stadtmöbel” n.d.). Straßenlaternen, Parkbänke oder Mülleimer können als Stadtmöbel betrachtet werden. Der technologische Fortschritt begünstigt derzeit die Entwicklung vom urbanen Mobiliar hin zu „Smart Street Furniture“, d.h. zunehmend intelligenten und vernetzten Stadtmöbeln.

Damit solches Mobiliar adäquat vom Menschen benutzt werden kann, sind geeignete Benutzerschnittstellen nötig. Nachfolgend werden einige Entwicklungen aus dem Bereich Smart Street Furniture mit Hinsicht auf ihre Interaktionsmöglichkeiten vorgestellt.

2.2.1.1. Straßenbeleuchtung

Reinhard Müllner und Andreas Riener entwickelten ein „Smart Street Lighting“ (SSL) System mit dem Ziel einer energieeffizienteren Straßenbeleuchtung (Müllner & Riener, 2011). Energieeffizienz soll u.a. erreicht werden, indem Straßenlaternen bei Bedarf durch Fußgänger eingeschalten und sonst ausgeschalten werden. Das SSL-Konzept sieht dabei eine Vernetzung der Straßenlaternen mit dem Smartphone der Fußgänger über einen Server vor. Das Smartphone übermittelt regelmäßig die aktuellen GPS-Koordinaten des Fußgängers an die SSL-Server. Der Server wiederum weist die Straßenlaternen in der unmittelbaren Umgebung an, sich zu erhellen. Zugleich werden Laternen, von welchen sich der Fußgänger entfernt, abgedunkelt. Über das Smartphone können Benutzer außerdem ihre gewünschte Safety Zone einstellen, also den Radius des Gebiets, das um sie herum erleuchtet werden soll (Müllner & Riener, 2011).

Durch SSL passt sich die Umgebung des Benutzers automatisch an dessen Bedürfnisse an. Die Interaktion zwischen Mensch und System ist dabei implizit, das System erwartet kein weiteres Zutun des Nutzers.

2.2.1.2. Interaktive Informationsbildschirme

Öffentliche Informationsbildschirme (engl. Public Displays) verbreiten sich in Städten zunehmend, Fortschritte in Display-, Kommunikations- und Interaktionstechnologien tragen dazu bei (Kostakos & Ojala, 2013). Dabei haben sich verschiedene Nutzungsszenarien und Interaktionsmethoden entwickelt.

Informationsdisplays werden bspw. in Bahnhöfen zur Wiedergabe von Abfahrtszeiten und Verkehrsinformationen eingesetzt (Mathew & Punnen, 2005). In diesem Szenario wird dem Betrachter keine Möglichkeit der Interaktion gegeben.

(Multi-)Touch-Displays bieten eine Benutzerschnittstelle, um Menschen die Interaktion mit öffentlichen Informationsbildschirmen zu gestatten. CityWall (Peltonen et al., 2008) ist ein öffentliches Multi-Touch- und Multi-User-Display, das in der Innenstadt von Helsinki installiert wurde. Es zeigt bspw. Fotos der Stadt, die in Echtzeit von Flickr geladen werden. Durch direkte Manipulationen über das Touch-Display können die Fotos verschoben, skaliert oder gedreht werden (Peltonen et al., 2008).

Brignull & Rogers haben jedoch herausgefunden, dass Menschen der Umgang mit solchen Displays in der Öffentlichkeit sozial unangenehm sein kann (Brignull & Rogers, 2003). Alternative Benutzerschnittstellen könnten dem entgegenwirken. In (Schroeter & Ronald, 2012) wird ein Konzept vorgestellt, bei dem Nutzer durch ihr Mobiltelefon mit einem öffentlichen Bildschirm interagieren. Der Inhalt der SMS wird zusammen mit einer (anonymen) Benutzerkennung auf dem Bildschirm angezeigt. Bluetooth- oder Wifi-fähige Mobiltelefone erlauben vielfältige weitere Interaktionsmöglichkeiten (Ballagas, Rohs & Sheridan, 2005; Peltonen et al., 2007).

2.2.1.3. Parkbänke

In „Adding Playful Interaction to Public Spaces“ werden zwei Projekte über interaktive, spielerische Sitzgelegenheiten vorgestellt (Dekel et al., 2005, pp. 226–228).

Die „Musical Chairs“ sind eine Reihe fest installierter Sitzblöcke, die akustische und Licht-basierte Signale abgeben können. Setzt sich eine Person auf einen davon, sendet dieser einen audiovisuellen Impuls aus. Setzt sich eine weitere Person auf einen anderen Block, so erzeugen die dazwischenliegenden Blöcke eine wellenartige Animation aus Tönen und Lichtern. Weitere Personen können durch Setzen auf weitere Blöcke die Welle unterbrechen oder verändern. Durch dieses Projekt sind Menschen sich spielerisch nähergekommen.

„The Intimate Bench“ ist eine Parkbank, die mit Sensoren und Lämpchen ausgestattet ist. Sie erkennt, wenn Menschen auf ihr sitzen und welche Entfernung zwischen ihnen liegt. Ist die Distanz zwischen ihnen groß, erzeugen die Lämpchen Muster, etwa Herzen, um die Personen anzuregen miteinander zu kommunizieren und das „soziale Eis“ zu brechen.

2.2.1.4. Mülleimer

Zunehmend verbreiten sich in Städten Mülleimer der Firma Bigbelly[10]. Das Unternehmen hat Mülleimer für den öffentlichen Raum entwickelt, die vernetzt sind, Füllstände erfassen und ihre Energie aus integrierten Solarpanelen beziehen. Nähert sich der Füllstand dem Maximalvolumen, wird automatisch eine Meldung an den Betreiber versandt. Die Geräte bilden damit ein Beispiel für ubiquitäres Computing und implizite Interaktion. Die Person, die Müll einwirft, ist in der Situation von einem unsichtbaren Computer umgeben. Dabei führt sie eine natürliche Aktion aus, die der Computer wahrnimmt und die implizit Prozesse auslösen kann.

2.2.1.5. Fußgängernavigation

Für Menschen mit Sehbeeinträchtigung ist es häufig schwer sich im öffentlichen Raum zu orientieren und Hindernisse auf ihren Wegen wahrzunehmen. In einem Versuch wurden Bluetooth iBeacons und Smartphones verwendet, um Sehbeeinträchtigte über ihre Umgebung zu informieren (van der Bie et al., 2016). Die iBeacons wurden in etwa zwei Metern Höhe an Straßenlaternen und Verkehrsschildern angebracht. Näherte sich eine Person mit ihrem Smartphone dem iBeacon, wurden Informationen zum Ort oder zu möglichen Hindernissen auf dem Display angezeigt und ggf. per Screen Reader vorgelesen.

2.2.1.6. Sprechende städtebauliche Objekte

Das Konzept „Hello Lamp Post“ der britischen Agentur PAN[11] besteht aus sprechenden, städtebaulichen Objekten. Personen können mit ihnen interagieren, indem sie SMS an eine ausgeschilderte Telefonnummer senden, die einen ebenfalls ausgeschilderten numerischen Code enthalten. Die Objekte reagieren auf die Nachrichten und beginnen Fragen auszusprechen, die ebenfalls per SMS beantwortet werden können und mit jeder Antwort persönlicher werden. Abbildung 2 illustriert das Szenario[12].

Abbildung 2 - Hello Lamp Post

2.2.1.7. Smart City

Unter dem Überbegriff der Smart City werden diverse Entwicklungen gesammelt, die darauf abzielen Städte durch Technologie effizienter und nachhaltiger zu gestalten. Intellistreets ist ein Konzept der britischen Firma Illuminating Concepts[13] über intelligente und multifunktionale Straßenlaternen. Die Lampen offerieren vielfältige Möglichkeiten: farbliche LED-Indikatoren für Routen oder Verkehrsbenachrichtigungen, integrierte Lautsprecher für Ankündigungen oder Hintergrundmusik, ein Bildschirm für digitale Verkehrsschilder, Werbung oder Hinweise, Rufen

---

[10] http://de.bigbelly.com/solutions/stations/ (letzter Abruf: 31.05.2017)

[11] http://panstudio.co.uk/project/hello-lamp-post/ (letzter Abruf: 14.08.2017)

[12] Bildquelle: http://panstudio.co.uk/wp-content/uploads/2015/07/hlp-pb-2.jpg (letzter Abruf: 14.08.2017)

[13] https://intellistreets.com/ (letzter Abruf: 14.08.2017)

von Hilfe in Notsituationen, Bilderkennung zur Verkehrsdatenerhebung, sowie Umweltsensoren zur Messung verschiedener Umgebungsparameter. Abbildung 3 zeigt eine Illustration einer solchen Lampe[14]. Erkennbar ist die eigentliche Laterne, darunter ein LED-Farbindikator, ein digitales Straßenschild, ein digitaler Informationsbildschirm, sowie eine Lampe zur Fassadenbeleuchtung. Die Laternen sind untereinander vernetzt.

Abbildung 3 - Intellistreets

2.2.2. Weitere interaktive Objekte im öffentlichen Raum

Im öffentlich-urbanen Raum wurden weitere Möglichkeiten interaktiver Objekte und Installationen erforscht, die vielfältige Interaktionen erlauben, auf Grund ihrer heterogenen Funktionen hier aber nicht weiter kategorisiert werden. Darunter sind z.B. Installationen von Medienfassaden oder lose, bewegliche Objekte.

2.2.2.1. Medienfassaden

Medienfassaden sind Anzeigen, die in die erbaute Umgebung (Gebäude, Straßenmöbel, etc.) integriert sind (Dalsgaard & Halskov, 2010), und damit eine eher seltene Kombination aus Architektur und Informationstechnologie darstellen (P. T. Fischer, Zollner, Hoffmann, Piatza, & Hornecker, 2013). Meist sind Medienfassaden temporäre Projektionen auf Fassaden, die häufig nur Werbung zeigen oder künstlerisch die Gestalt von Gebäuden verändern, selten jedoch mit sich interagieren lassen (P. T. Fischer et al., 2013).

Patrick T. Fischer beschäftigt sich u.a. mit multimodalen Interaktionsmöglichkeiten für öffentliche Medienfassaden und hat mit dem Projekt SMSlingshot eine auffällige Variante entwickelt. Das Projekt besteht aus einer projizierten Medienfassade und einem Steinschleuder-ähnlichen Handgerät mit Tastatur und Display. Personen können in das Gerät Nachrichten eintippen und diese durch Aufziehen und Loslassen der Schlinge an die Fassade schleudern. Dort erscheinen dann Farbklekse mit der getippten Nachricht (Patrick Tobias Fischer, Hornecker & Zoellner, 2013).

2.2.2.2. Bewegliche Objekte

Bewegliche, interaktive Objekte im urbanen Raum bieten neue Möglichkeiten, stellen aber auch neue Herausforderungen. Projekte wie PlazaPuck (Patrick Tobias Fischer, Hornecker, Umar, & Anusas, 2013), Movable, Kick-/Flickable Light Fragments (Patrick Tobias Fischer et al., 2014) und (Seitinger, Taub, & Taylor, 2010) untersuchen dies. Objekte sind dabei bspw. Erdnuss- oder Birnenförmige kleine, leuchtende Geräte, die auf Berührungen oder Töne reagieren. Die Light Fragments von Fischer lassen sich bspw. treten, die Light Bodies von Seitinger reagieren auf Musik.

[14] Bildquelle: https://intellistreets.com/images/splash.jpg (letzter Abruf: 14.08.2017)

2.3. Internet of Things

Cáceres und Friday untersuchten 20 Jahre nach Weisers UbiComp-Vision, inwieweit die Vision bereits Einzug in die Realität hielt und welche Möglichkeiten es gibt, diesen Prozess weiter zu fördern (Caceres & Friday, 2012). Dabei identifizierten sie zwei wesentliche Technologien als Treiber für den Aufbau einer UbiComp-Infrastruktur: Cloud Computing und das Internet of Things (kurz IoT).

2.3.1. Grundlagen

In einem Report an die europäische Kommission wurde das Internet of Things als weltweites Netzwerk verbundener, einzigartig adressierbarer Objekte auf Basis von Standardkommunikationsprotokollen bezeichnet (Botterman, 2009, p. 5).

Nach Gubbi et al. bestehe IoT aus zusammenwirkenden, fühlenden und auslösenden Geräten, die Informationen über Plattformen teilen und damit innovative Anwendungen ermöglichen (Gubbi, Buyya, Marusic, & Palaniswami, 2013, p. 4).

Um sich dem Bewusstsein der Nutzer zu entziehen, fordere IoT drei Dinge: „(1) ein gemeinsames Verständnis über die Situation seiner Nutzer und seiner Anwendungen, (2) Softwarearchitekturen und übergreifende Kommunikationsnetzwerke, um kontextbezogene Informationen dort zu verarbeiten und zu übermitteln, wo sie relevant sind, und (3) Analysewerkzeuge in IoT mit dem Ziel des autonomen und smarten Verhaltens“ (Gubbi et al., 2013, pp. 1–2).

Nachfolgend werden einige Konzepte des IoT näher beschrieben.

2.3.2. Elemente

Gubbi et al. erläutern, dass das IoT aus den Komponenten Hardware (Sensoren, Auslöser und integrierte Kommunikation), Middleware (Persistenz und Computing) und Darstellung (Visualisierung und Interpretation) bestehe. Darüber hinaus identifizieren sie folgende Technologien, die für das IoT Schlüsselfunktionen bieten.

Etwa RFID-Tags, kleine Chips ohne dauerhafte Stromversorgung, könnten an Objekten angebracht und von RFID-Lesern identifiziert werden. Sie spielen bereits eine Rolle im Warenhandel und in der Logistik, sind im Sinne von IoT jedoch passive Objekte.

Wireless Sensor Networks (kurz WSN) hingegen bestehen aus kleinen, günstigen und energiesparenden Sensorgeräten, die drahtlos an ein Netzwerk angebunden sind. Die erhobenen Sensordaten werden verteilt oder zentral für Analysen ausgewertet (Gubbi et al., 2013, pp. 5–6). WSN bestehen nach Gubbi et al. aus der Hardware der Sensorknoten, einem Kommunikationsstapel, Middleware, sowie sicherer Datenaggregation.

Adressschemen seien notwendig, um jedes Gerät einzigartig, verlässlich, dauerhaft und skalierbar identifizieren zu können. IP-Adressen (IPv4 oder IPv6) könnten diese Kriterien nicht ausreichend erfüllen.

Eine zunehmend wichtige Rolle spiele auch Middleware, die technologische und anwendungsbezogene Schichten voneinander trennt (Atzori, Iera, & Morabito, 2010, p. 2791). Die Entwicklung von IoT-Anwendungen werde damit erleichtert, da die Details tieferer, technischer Schichten versteckt werden. Der gängige Ansatz serviceorientierter Architekturen werde auch für viele IoT-Szenarien empfohlen. Eine Middleware-Komponente kann in einem

WSN bspw. ein Knoten sein, der drahtlos mit Sensoren in geringer Reichweite und mit geringer Bandbreite in Verbindung steht und ihre Daten über TCP/IP an die Cloud weitersendet.

2.3.3. Architekturen

Sowohl in herkömmlichen Unternehmensinformationssystemen, als auch in IoT-Szenarien haben sich Architekturparadigmen entwickelt, um Anforderungen wie Erweiterbarkeit, Zuverlässigkeit und Wartbarkeit von System zu gewährleisten. Die hier erläuterten, gängigen Konzepte sind nicht zwingend IoT-spezifisch.

2.3.3.1. Serviceorientierte Architektur

Serviceorientierte Architekturen (kurz SOA) verfolgen das Ziel, komplexe und monolithische Systeme in einfache und wohldefinierte Komponenten aufzugliedern. Die Serviceorientierung bedeutet, dass Services die zentrale Abstraktionseinheit bilden und mittels Komposition zu Prozessen und Abläufen zusammengestellt werden (Atzori et al., 2010, pp. 2791–2792). Atzori et al. schlagen daher eine die Serviceorientierung-unterstützende Middleware-Schicht vor. Dabei seien Service-Komposition, Service Management und Objektabstraktion die zentralen Zuständigkeiten der Middleware. Für Informationssysteme bietet die Business Process Execution Language (kurz BPEL) eine verbreitete Methode der Service-Komposition. Im Fall von Webservices können Entitäten und Schnittstellen bspw. mit Hilfe von WSDL beschrieben und durch SOAP übertragen werden.

Representational State Transfer[15] (kurz REST) ist ein Architekturkonzept, das erstmals von Roy Fielding vorgestellt wurde (Fielding, 2000). Es lässt sich ebenfalls für Service-basierte Implementierungen einsetzen, erfüllt die Anforderungen einer SOA im allgemeinen Konsens jedoch nicht gänzlich. REST ist kein Standard und anders als SOAP umfasst es keine fixen Schnittstellendefinitionen. REST wird oft mit HTTP und URIs angewendet. Dabei ist eine URI die Adresse einer Ressource auf einem Server, mit der gearbeitet werden kann. Meist sind die Arbeitsbefehle die Standard-HTTP-Verben, wie GET, POST, PUT und DELETE. Die Verben lassen ein einfaches CRUD-Muster (Create, Read, Update und Delete) erkennen. Daran ist auch bemerkbar, dass REST im Gegensatz zum serviceorientierten RPC oder SOAP einen eher ressourcenorientierten Ansatz verfolgt. Teils wird daher auch der Begriff Ressourcenorientierter Architekturen (kurz ROA) verwendet. Darüber hinaus ist REST zustandslos, d.h. es werden serverseitig keine Sessions o.ä. über mehrere Anfragen hinweg aufrechterhalten.

2.3.3.2. Eventgetriebene Architektur

SOA als Architekturmuster basiert auf dem Wissen über Geschäftsprozesse und weist Schwächen auf, wenn dieses nicht gänzlich bekannt ist oder sich oft verändert. Es fehlt Flexibilität für neu hinzukommende oder gar intelligente, dynamische Abläufe. In vielen SO-Architekturen überwiegen außerdem synchrone Aufrufe als Invokationsmethode. Dies kann Latenzen verursachen, die für Echtzeitanwendung kritisch sind (Bruns & Dunkel, 2010).

Eventgetriebene Architekturen (kurz EDA) verfolgen einen Entwurfsstil, der sich komplementär zu SOA anwenden lässt. Im Zentrum stehen dabei Ereignisse, die auf unterschiedlichste Weise entstehen können. Ereignisse können von Nutzern, wie auch von Prozessen oder anderen externen Umständen ausgelöst werden. Das Modell kennt dabei die Elemente Ereignisquelle,

[15] https://de.wikipedia.org/wiki/Representational_State_Transfer (letzter Abruf: 13.08.2017)

Ereignissenke und Ereignisobjekt. Die Quelle produziert Ereignisnachrichten auf Basis von Informationen, die ihr zur Verfügung stehen. Die Senke empfängt solche Nachrichten und verarbeitet sie, sobald sie eingegangen sind. Das Ereignisobjekt bzw. die Ereignisnachricht ist der Gegenstand, der übertragen wird. Es enthält keinerlei Informationen über Verarbeitungen, die eine Senke damit in Gang setzt. Es ist lediglich die Information, dass ein Ereignis eingetroffen ist. Das Kommunikationsmuster in EDA ist dabei normalerweise asynchron. Nachrichten werden nicht direkt ausgetauscht, sondern über einen Mediator verteilt. Eine Nachricht kann auch mehrere Senken erreichen. Es lässt sich hier nicht nur von einer starken Entkopplung von Komponenten (Quelle und Senke) reden: Quellen und Senken sind einander unbekannt. Gängig jedoch ist, dass eine Komponente zugleich Quelle und Senke sein kann, also Ereignisse sendet und empfängt. Die typische Implementierung einer EDA nutzt ein Publish/Subscribe-Modell für die Mediation von Nachrichten (Bruns & Dunkel, 2010, pp. 47–56).

2.3.3.3. Cloud Computing

Cloud Computing[16] (kurz CC oder nur Cloud) ist ein Modell verteilter und skalierbarer On-Demand-Rechenressourcen, die über ein Netzwerk erreicht werden können. Internetkonzerne wie Google und Facebook haben maßgeblich zu den Technologien beigetragen. Die Amazon Web Services gelten als größter Anbieter von CC. Es ist geeignet riesige Datenmengen zu verarbeiten und sich Auslastungen anzupassen. In 2.3 wird darauf hingewiesen, dass Cáceres und Friday Cloud Computing als Schlüsselfaktor für den Erfolg von UbiComp sehen.

Aus Sichtweise dieser Arbeit ist CC eine Rechenzentrum-seitige Disziplin. Wenn auch CC architektonische Einflüsse auf IoT haben dürfte, wird die Cloud in dieser Arbeit mit der Bedeutung einer serverseitigen Backendkomponente abstrahiert, auf dessen Interna nicht näher eingegangen wird.

2.3.3.4. Fog Computing

Entgegen der Einschätzung von Cáceres und Friday gibt es auch Einschätzungen, dass CC nicht für jedes IoT-Szenario geeignet ist. Echtzeitanwendungen, etwa für vernetzte, autonome Fahrzeuge, erfordern eine schnelle Datenverarbeitung und –übertragung. Der Weg quer durch das Internet und durch die Cloud kann hohe Latenzen erfordern.

Fog Computing (kurz FC oder nur Fog) ist ein Ansatz, der sich dem CC-Paradigma entgegenstellt und fordert, Datenverarbeitung wieder dezentral und näher am ‚Boden' des Internets durchzuführen, also in geographischer oder netzwerktopologischer Nähe zu den Geräten, die die Verarbeitung erfordern. Gedacht ist dabei etwa zusätzliche Middleware, die die Anwendung oder Teile davon ausführen kann. Im Beispiel des vernetzten, autonomen Fahrens könnte solche Middleware an Autobahnen eingesetzt werden (Bonomi, Milito, Zhu, & Addepalli, 2012).

Im extremsten Fall des Fogs wird die Anwendung auf den Endgeräten selbst ausgeführt. Der Begriff Edge Computing gewinnt in diesem Kontext an Bedeutung.

---

[16] https://de.wikipedia.org/wiki/Cloud_Computing (letzter Abruf: 13.08.2017)

2.3.4. Bluetooth LE & iBeacons

Bluetooth Low Energy (kurz LE), auch genannt Bluetooth Smart, wurde 2010 als Teil des Standards Bluetooth 4.0 von der Bluetooth Special Interest Group eingeführt[17]. Die Besonderheit gegenüber dem herkömmlichen Bluetooth-Standard ist, dass BLE deutlich weniger Energie benötigt und dadurch auch in sehr kleinen Geräten verbaut werden kann – zu Lasten der maximalen Übertragungsrate. Produkte, die mit „Bluetooth Smart Ready" gekennzeichnet sind, unterstützen sowohl den klassischen Bluetooth Standard, als auch LE-Geräte.

Bei Bluetooth LE Geräten lässt sich zwischen den Rollen „Central" und „Periphal" unterscheiden[18]. Um zwei Geräte miteinander zu verbinden, ist stets je ein Gerät jeder Rolle notwendig. Das Periphal bewirbt sich selbst durch das Ausstrahlen von Werbungspakaten (engl. Advertisement Packets). Das Central initiiert das Scannen nach solchen Werbungspaketen, horcht also danach. Wird ein Periphal gefunden, kann das Central die Verbindungsaufnahme beginnen. Das Protokoll organisiert sich in Services, die ein Periphal anbietet. Services wiederum weisen Characteristics auf, die ausgelesen und beschrieben werden können.

2013 stellte Apple iBeacons vor. Die Technologie basiert auf Bluetooth LE und nutzt lediglich dessen Advertising-Mechanismus. Wie Leuchtfeuer (engl. Beacons) senden diese Geräte kurze Nachrichten an ihre Umgebung aus. Der Nachrichteninhalt besteht lediglich aus einer 16 Byte langen UUID, einem zwei Byte Major und einem zwei Byte Minor. Anhand dem Received Signal Strength Indicator (kurz RSSI) können empfangende Geräte die Entfernung zum iBeacon

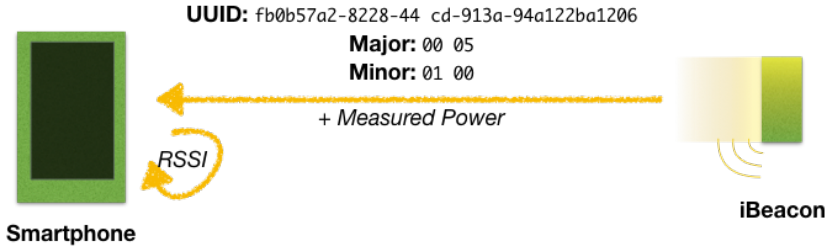

Abbildung 4 - iBeacon-Übertragung

approximieren. Um einen Anhaltspunkt für die Distanzberechnung zu erhalten, senden die iBeacons einen RSSI-Referenzwert mit, der bei einer Entfernung von einem Meter zu erwarten ist. Abbildung 4 illustriert die Funktionsweise eines iBeacons.

iBeacons werden häufig für In-House-Navigation oder ortsabhängige Werbung genutzt. Dabei horchen z.B. Smartphones auf iBeacons und lösen Events in Apps aus, falls welche gefunden werden. UUID, Major und Minor können die Apps nutzen, um sich weitere Informationen einzuholen (z.B. von Webservices, nicht vom iBeacon selbst) und auf etwas hinzuweisen. Dabei sollten alle iBeacons des gleichen Anwendungsfalls die gleiche UUID aufweisen. Mit Major und Minor können sie dann innerhalb dieser Anwendung identifiziert werden. Bspw. könnte eine Supermarktkette iBeacons für Werbe- und Navigationszwecke einsetzen. Es hätten alle iBeacons

---

[17] https://en.wikipedia.org/wiki/Bluetooth_Low_Energy (letzter Abruf: 28.07.2017)

[18] https://developer.apple.com/library/content/documentation/NetworkingInternetWeb/Conceptual/CoreBluetooth_concepts/AboutCoreBluetooth/Introduction.html ff. (letzter Abruf: 28.07.2017)

dieser Kette die gleiche UUID. Die einzelnen Geräte lassen sich dann z.B. über die Filialnummer (Major) und die Regalnummer (Minor) identifizieren.

Das iBeacon-Protokoll ist rein unidirektional und bietet keine weiteren Möglichkeiten für Informationsaustausch. Kombinationen mit dem restlichen Bluetooth LE Standard für mehr Möglichkeiten der Datenübertragung sind denkbar, scheinen in der Praxis aber eher unüblich. Bluetooth LE bietet außerdem nicht den Grad an Verschlüsselungs- und Abhörsicherheit (Ryan, 2013), den man sich für die Übertragung vertraulicher oder persönlicher Daten wünscht.

2.3.5. MQTT

MQ Telemetry Transport[19] (kurz MQTT) ist ein Standard für ein Publish/Subscribe-Protokoll, das für und unter den Gesichtspunkten des Internet of Things entwickelt wurde. Es ist ausgelegt für Umfelder, in welchen Ressourcen wie Bandbreite und Speicher limitiert sind. Es besteht aus den Komponenten Subscriber, Publisher und Broker. Publisher erzeugen interessante Nachrichten zu Themen (Topics), Subscriber abonnieren Topics und erhalten die Nachrichten, die dazu veröffentlicht wurden. Der Broker nimmt Nachrichten von Publishern entgegen und verteilt sie je nach Abonnements an die Subscriber. Der Broker verfügt über Fähigkeiten, um Ressourcen (Topics) mittels Autorisierung gegenüber Subscribern und Publishern abzusichern (Al-Fuqaha, Guizani, Mohammadi, Aledhari, & Ayyash, 2015, pp. 2354–2355).

[19] http://mqtt.org/ (letzter Abruf: 06.08.2017)

3. Anforderungen

Das Ergebnis dieser Arbeit soll gewissen Ansprüchen genügen, zugleich setzt es aber auch Anforderungen voraus. In den folgenden Abschnitten wird erklärt, welche Kriterien das Konzept über Mikroinformationsstrahler erfüllen soll und welche Umstände im Konzept als gegeben betrachtet werden. Ist in diesem Kapitel von Anforderungen an und vom Konzept die Rede, so bezieht sich der Begriff i.d.R. nicht nur auf Kapitel 4 über das Konzept, sondern auch auf Kapitel 5 über den Prototyp.

3.1. Kontext

Dieses Konzept über Mikroinformationsstrahler wird für den Kontext des Projekts UrbanLife+ (nachfolgend kurz UL+) entwickelt. Das Projekt zielt insbesondere darauf ab, Seniorinnen und Senioren bei ihrer Teilhabe im urbanen Raum zu unterstützen und zu Aktivitäten anzuregen. In UL+ werden verschiedene Komponenten und Systeme entwickelt und angebunden. Diese werden im Folgenden kurz erläutert und Begrifflichkeiten für das Konzept definiert.

3.1.1. Informationsbildschirme

Große Informationsbildschirme können viele Informationen strukturiert wiedergeben und erlauben durch Touch-Fähigkeit vielfältige Interaktionsmöglichkeiten. In UL+ übernehmen sie eine zentrale Funktion. Senioren können sich auf Bildschirmen Umgebungsinformationen zu Orten wie Restaurants, Parks und ÖPNV ansehen, Routen planen oder sich persönlichen Herausforderungen stellen. Dabei verbindet sich der Bildschirm automatisch mit dem Smartphone des Nutzers und zeigt individualisierte Inhalte an. Solche Bildschirme können bspw. in Seniorenheimen oder Banken installiert werden. Neben großen Wandbildschirmen soll es auch sog. Miniinformationsstrahler geben, die mit einem kleineren Bildschirm im Tablet-Größenfaktor ausgestattet sind. Als Installationsort sind hier etwa Bushaltestellen, Geschäfte oder Sitzbänke denkbar. Miniinformationsstrahler sind in ihren Fähigkeiten gegenüber ihrem größeren Pendant etwas eingeschränkt, dafür aber kostengünstiger und etwas einfacher zu installieren.

Abbildung 5 - Bildschirmfoto: UrbanLife+ Informationsbildschirm

Abbildung 5 zeigt einen aktuellen Arbeitsstand der Entwicklung der UrbanLife+-Informationsbildschirme. Zu sehen ist eine Vielzahl sich bewegender Komponenten mit Inhalten, mit welchen interagiert werden kann.

Basis für die Informationsbildschirme ist das Projekt CommunityMirrors der Forschungsgruppe Kooperationssysteme der Universität der Bundeswehr München. In dieser Arbeit werden die Begriffe Informationsbildschirm, Informationstafel, CommunityMirror, Makro- und Miniinformationsstrahler teils synonym verwendet. Darüber hinaus ist angedacht, dass sich Informationsbildschirme und Mikroinformationsstrahler (kurz MIS) gemeinsamen Nutzungsszenarien fügen und entsprechend zusammenarbeiten. Ist vom Begriff Informationsstrahler (kurz IS) die Rede, sind i.d.R. sowohl Bildschirme, als auch Mikrostrahler gemeint.

3.1.2. Persönliche Endgeräte

Um Nutzer gegenüber bspw. Informationsbildschirmen oder Informationsstrahlern zu authentisieren, sollen persönliche Endmittel zum Einsatz kommen. Dabei bieten sich besonders Smartphones mit Bluetooth Low Energy (LE) Schnittstelle an. Über die iBeacon-Technologie können Smartphones IS automatisch und drahtlos erkennen und über einen geeigneten Kommunikationskanal eine Aktion auslösen. Über Bluetooth ist bspw. weiterer Datenaustausch möglich, etwa der Transfer von persönlichem Profil und Einstellungen. Voraussetzung für dieses Konzept ist eine mobile App mit entsprechender oder ähnlicher Funktionalität, die auf dem Smartphone installiert ist.

Im Kontext der UL+-Projektpartner werden auch schon persönliche Kundenkarten im Scheckkartenformat eingesetzt. Die Informationsauslese könnte dann etwa über einen Magnetstreifen oder drahtlos über RFID stattfinden. Auf Grund der Flexibilität und Verbreitung setzt dieses Mikroinformationsstrahler-Konzept jedoch hauptsächlich auf den Ansatz mit Smartphone. Eine entsprechende iOS-App existiert bereits als Prototyp zum Testen der Funktion.

3.1.3. CommunityMashup

CommunityMashup ist ebenfalls ein Projekt der Foschungsgruppe Kooperationssysteme der Universität der Bundeswehr München. Dabei handelt es sich um einen Webservice, der Informationen aus verschiedenen Quellen sammelt, aggregiert, vereinheitlicht und zur Verfügung stellt. Zentrale Informationsobjekte sind dabei Personen, Organisationen und Inhalte. Quellen können bspw. soziale Netzwerke oder andere externe Dienst- und Ressourcenanbieter sein.

Für die aktuellen Entwicklungen der IS-Komponenten ist das CommunityMashup die Hauptdatenquelle. Es versorgt mit Nutzerstamm und -bewegungsdaten wie auch mit Informationen über Orte und Organisationen. Auch für dieses Konzept ist es von zentraler Bedeutung. Der Service existiert bereits und stellt eine REST-API zur Verfügung, über welche die Informationsobjekte und ihre Beziehungen abgefragt werden können.

An dieser Stelle ist zu sagen, dass im UL+-Projekt auch andere serverseitige Komponenten in Entwicklung sind oder noch entwickelt werden, die ähnliche oder ergänzende Funktionen bereitstellen. In dieser Arbeit wird der Begriff Cloud als Abstraktion der verschiedenen Backend-Komponenten genutzt. Teils wird auch der Begriff CommunityMashup synonym verwendet. Zum Zeitpunkt der Verfassung dieser Arbeit sind die genauen Zuständigkeiten der serverseitigen Implementierungen noch nicht gänzlich geklärt.

3.1.4. Orte und Dienstleistungen

In UL+ sollen Informationen zu Orten nicht nur wiedergegeben, sondern Orte und Dienstleistungen aktiv in das Gesamtkonzept integriert werden. Das bedeutet z.B., dass eine Bäckerei nicht nur in der Karte eines Informationsbildschirms angezeigt wird, sondern auch Teil einer Herausforderung ist oder Vergünstigungen als Belohnung für das Erreichen eines Ziels bietet. Dabei sind vielfältige Anbindungen von Friseurgeschäften über ÖPNV hin zu Museen denkbar. Die UL+-Cloud wird hier eine Lösung bieten, um solche Orte und Dienstleistungen als Quellen zu integrieren.

3.1.5. Weitere

Teil von UL+ sind auch viele andere Unterprojekte über bspw. intelligente Straßenbeleuchtung, Parkbänke oder den Safety-Atlas. Im Safety-Atlas sollen sicherheitsrelevante Informationen über den öffentlichen Raum gesammelt werden. Die Daten können nicht nur von städtischen Planern und Entscheidern gebraucht werden, sondern auch in UL+, um Senioren etwa auf barrierefreie Routen hinzuweisen und sie damit bestmöglich zu unterstützen. Federführend ist hierbei Drees & Sommer Infra Consult und Entwicklungsmanagement GmbH aus Stuttgart. Genauere Spezifikationen oder prototypische Umsetzungen sind derzeit nicht bekannt.

3.2. Szenario

Die Basis des Anwendungsszenarios bilden die internen Projektdokumente „FuE-Vorhaben Informationsstrahler 1.1“ und „FuE-Vorhaben Informationsstrahler - MTI Gestaltung 1.2“ der Forschungsgruppe Kooperationssysteme an der Universität der Bundeswehr München. Die Wesentlichen der darin enthaltenen Ideen für das UL+-Konzept werden nachfolgend zusammengefasst. Die fiktive Handlung des folgenden Szenarios ist eine Zusammenstellung aus beiden Dokumenten und teilweise inhaltlich aufbereitet und ergänzt. Der Einsatz von Mikroinformationsstrahlern wird dabei ausgeblendet, um die Notwendigkeit der Strahler und die Anforderungen an die Strahler anschließend herauszustellen. Es werden auch die ersten Gedanken zum Mikroinformationsstrahler aus den Projektdokumenten dargelegt. Ziel dieser Arbeit ist es, das Konzept über Mikroinformationsstrahler in dieses Szenario einzubetten und dessen Anforderungen gerecht zu werden.

Das Szenario beschreibt die 82-jährige Seniorin Margot als Persona, die sich selbstständiger und sicherer in der Stadt Mönchengladbach bewegen möchte. Besonders in Gegenden, die sie nicht gut kennt, fühlt sie sich unsicher. Dennoch möchte sie gerne den neu eröffneten Hardterbroicher Markt besuchen.

Es wird angenommen, dass Mönchengladbach mit technischer Infrastruktur für die Anwendung von UL+ ausgestattet ist. Ferner hat Margot ein Bluetooth-fähiges Smartphone mit installierter UL+-App und sei bei der UL+-Informationsplattform registriert und angemeldet.

> Margot geht nach dem Frühstück zur digitalen, interaktiven Informationstafel im Eingangsbereich ihrer Seniorenwohnanlage, um sich zu informieren, was sie heute in der Stadt unternehmen könnte. Angezeigt werden ihr etwa Veranstaltungen, Orte und Organisationen in der Umgebung. Als sie sich dem Bildschirm nähert verbindet sich dieser automatisch mit dem Smartphone in ihrer Tasche. Ohne weiteres Zutun überträgt das Smartphone Angaben über Margot, darunter Name, körperliche Einschränkungen und Informationen zu Margots Komfortzone. Die Komfortzone ist das geografische Gebiet,

indem sie sich besonders häufig aufhält und sich sicher fühlt. Die Zone verändert sich situativ und wird anhand ihrer Aktivitäten und mittels Positionserfassung laufend angepasst. Der Informationsbildschirm reagiert auf diese Daten und zeigt einen individuellen, persönlichen Bereich mit der Begrüßung „Hallo Margot!“ an. Außerdem wird ein Bestätigungston abgespielt. Margot hat eine leichte Sehschwäche, deshalb werden die Informationen in ihrem Bereich etwas größer angezeigt. In diesem Bereich findet sie Empfehlungen, die an ihre Bedürfnisse angepasst sind.

Margot sieht auf der Tafel, dass der Hardterbroicher Markt heute eröffnet wird. Sie hat schon davon gehört und berührt das zugehörige Symbol. Es werden nun zusätzliche Informationen dargestellt, etwa die Öffnungszeiten und das Programm der Eröffnungsfeier. Außerdem sieht Margot, dass ihre Freundin Brigitte heute ebenfalls auf den Markt geht. Sie entschließt sich den Markt zu besuchen, also berührt und schiebt sie das Symbol in ihren persönlichen Bereich. Es wird ihr nun zusätzlich angezeigt, welche ÖPNV-Verbindungen sie dorthin nutzen könnte, sowie wo und wann diese starten.

Margot entfernt sich von der Tafel - der persönliche Bereich verschwindet - und macht sich auf den Weg. In der näheren Umgebung der Wohnanlage kennt sie sich gut aus, die Bushaltestelle für ihre Abfahrt findet sie problemlos. An der Haltestelle sieht sie eine kleine weitere Informationstafel - ein Miniinformationsstrahler. Er erkennt Margot ebenfalls automatisch an ihrem Smartphone, zeigt ihr Informationen zur Abfahrt ihres Busses und an welcher Haltestelle sie aussteigen sollte. An der Zielhaltestelle kennt Margot sich noch bis hinter die Sparkasse aus. An der Sparkasse sieht sie eine weitere Informationstafel. Die Tafel verbindet sich mit Margots Smartphone und zeigt ihren persönlichen Bereich an. Als Anregung wird Margot vorgeschlagen heute am Hardterbroicher Markt ein Café zu besuchen. Sie hält dies für eine gute Idee und tippt auf die Herausforderung, um sie anzunehmen.

Da Margot sich bis zum Markt nicht besonders auskennt, sieht sie sich die Umgebungskarte auf dem Bildschirm genauer an und versucht sich die Route einzuprägen. Auf dem weiteren Weg zum Markt sieht sie vereinzelt Miniinformationsstrahler, auf welcher sie die Karte noch einmal ansehen kann. Da Margot ihre Komfortzone verlassen hat, zeigen die Strahler vermehrt in der Nähe verortete Sitzgelegenheiten und Notrufmöglichkeiten, sowie Richtungen in denen sich ebenfalls Personen befinden, deren Smartphones mit Strahlern verbunden sind und die in Notsituationen Hilfe rufen könnten. Margot fühlt sich sicherer, trotzdem muss sie unterwegs einmal nach dem Weg fragen. Schließlich gelangt Margot an ihrem Ziel an.

Auch Brigitte kann sie vor Ort finden. An einer weiteren Informationstafel informieren sich die beiden über Cafés in der Nähe. Der Bildschirm erkennt sowohl Brigitte als auch Margot und arrangiert die Inhalte am Bildschirm so, dass beide persönlichen Bereiche gleichzeitig angezeigt werden. Die beiden sehen, dass in einem Café ums Eck heute Donauwelle für NEW Kunden im Angebot ist, und beschließen dorthin zu gehen. Dabei erfüllt Margot ihre zuvor angenommene Herausforderung. Auch auf dem Heimweg unterstützen die Informationstafeln Margot. Da Margot heute viel in neuen Teilen der Stadt unterwegs war und sich dort etwas sicherer fühlt, wird ihre Komfortzone entsprechend angepasst.

Das Szenario zeigt, welche Unterstützung mittels kleiner und großer Informationstafeln möglich ist. Allerdings zeigt es auch, wo sich im Verlauf der Nutzung Lücken ergeben, die bei Margot für Unsicherheit und Orientierungslosigkeit sorgen. Die Tafeln sind recht teuer und aufwendig zu installieren, ihre Größen- und Preisfaktoren verhindern, dass Städte dicht mit ihnen bebaut werden. Dennoch wäre es wünschenswert, Margot kontinuierlich und überall in ihrem Alltag und auf ihren Wegen zu unterstützen, so lässt sich die Forderung nach vielen noch kleineren, kostengünstigeren Geräten begründen – den Mikroinformationsstrahlern.

Die Projektdokumente enthalten bereits einige Ideen zu solchen Strahlern, spezifizieren sie aber noch kaum. Zum Einsatz kommen sollen Licht- und Audio-Komponenten. Beschrieben werden bspw. „smarte Leuchten“, die bei Annäherung automatisch einen Licht- und Tonimpuls aussenden. Durch einen leuchtenden Pfeil weisen sie außerdem den Weg.

Die folgenden Abschnitte folgern aus dem Anwendungsszenario einige Anforderungen an die Mikroinformationsstrahler.

3.3. Kosten

Wenn auch diese Arbeit sich kaum näher mit der Kostenrechnung der UL+-Infrastruktur beschäftigt, ist anzunehmen, dass für den Aufbau eines dichten Netzes von Informationsstrahlern eine gewisse Kosteneffizienz notwendig ist. Betrachtet man das Szenario, lässt sich behaupten, dass eine höhere Dichte an Informationsstrahlern bessere Möglichkeiten bietet um Senioren zu unterstützen.

Um monetäre Kosten zu sparen, lassen sich bspw. die Kostenstellen Gerätepreis, Montagekosten, Wartungskosten und Versorgungskosten betrachten. Sie alle bieten ein erhebliches Potential. Der Gerätepreis kann deutlich sinken, wenn simple Hardwarekomponenten verwendet werden und auf teure Bauteile wie Bildschirme verzichtet wird. Die Montage und Installation kann bei kleinen Geräten u.U. deutlich günstiger sein, als etwa bei großen Wandbildschirmen. Ein durchdachtes Hard- und Softwarekonzept kann die Häufigkeit von benötigten Wartungsmaßnahmen reduzieren und energiesparende Hardwarekomponenten verringern die Stromkosten.

Das Konzept befasst sich nicht mit konkreten, finanziellen Zielen. Doch Kostenfaktoren sind mitunter die Motivation dieser Arbeit, weshalb die Fähigkeiten sehr einfacher interaktiver Geräte behandelt werden. Insofern ist die Kostenanforderung eher eine motivierende Richtlinie.

3.4. Mensch-Computer-Interaktion

Das in dieser Arbeit vorgestellte Konzept impliziert das Vorkommen von MCI. Ziel ist es Menschen im Alltag durch Technologie zu unterstützen. In dieser Arbeit soll sich jedoch lediglich mit besonders einfachen Benutzerschnittstellen beschäftigen: Licht- und Audio-Komponenten. Darüber hinaus ist der Einsatz der iBeacon-Technologie vorgesehen. Dabei kommen verschiedene explizite und implizite Interaktionsparadigmen zum Vorschein.

Es gibt diverse Evaluationsmöglichkeiten, um zu ermitteln, wie gut Interaktionskonzepte von Nutzern angenommen werden und wo es Verbesserungspotenzial gibt. Das vorgestellte Konzept bzw. die vorgestellte Szenario-Vision wurden primär auf Basis des kognitiven Durchgangs der Abläufe durch den Autor entwickelt. Direkter Austausch mit potenziellen Nutzern, also z.B. Senioren, durch Befragungen, Feldstudien oder Inspektionen sind nicht in diesem Rahmen inbegriffen.

Die drei Faktoren Effektivität, Effizienz und Zufriedenheit aus ISO 9241-11 bilden die Basis für die Überlegungen zum Interaktionskonzept. Sie sind gewissermaßen als Anforderungen an die Interaktion zu verstehen, werden in dieser Arbeit jedoch nicht durch Messungen o.ä. evaluiert. Auch Shneidermans acht Regeln und die in 2.1.4 vorgestellten universellen Designprinzipien sollen soweit möglich geachtet werden.

Darüber hinaus soll das Konzept Mehrbenutzerfähigkeiten aufweisen. Da die Informationsstrahler im urbanen Raum fest installiert sind und von jedem genutzt werden können, muss das Konzept hier entsprechende Möglichkeiten vorsehen.

3.5. Architektur

Aus dem Szenario und dem Kontext von UL+ ergeben sich einige Ansprüche an die Architektur und die Vernetzung der Komponenten. Informationen stammen aus einer Vielzahl heterogener Quellen und auch die Datenschnittstellen verfügen über unterschiedliche Fähigkeiten und Grenzen. Das klassische *Eingabe-Verarbeitung-Ausgabe*-Muster der Softwareentwicklung nimmt komplexe Gestalten an, wenn neben räumlichen, visuellen und akustischen Sensoren und Aktoren auch Ereignisse an anderen Informationsstrahlern zu den Ein- und Ausgabesignalen gezählt werden.

3.5.1. Komponenten

Ziel des Konzepts ist es, einige der wesentlichen Komponenten aus UL+ anzubinden. Aus Perspektive des Nutzers sollten insbesondere persönliche Endgeräte und das UL+-Informationsportal angebunden sein. Im Genaueren gehören z.B. das CommunityMashup und der Safety-Atlas zu diesem Portal. Die Funktion der Mikroinformationsstrahler ist außerdem durch die Integrität mit anderen Informationsstrahlern (Tafeln, Ministrahler) bestimmt. Die Strahler sollen zusammenarbeiten und den Nutzer als gemeinsame Einheit unterstützen. Abbildung 6 gibt einen Überblick über die verschiedenen Komponenten.

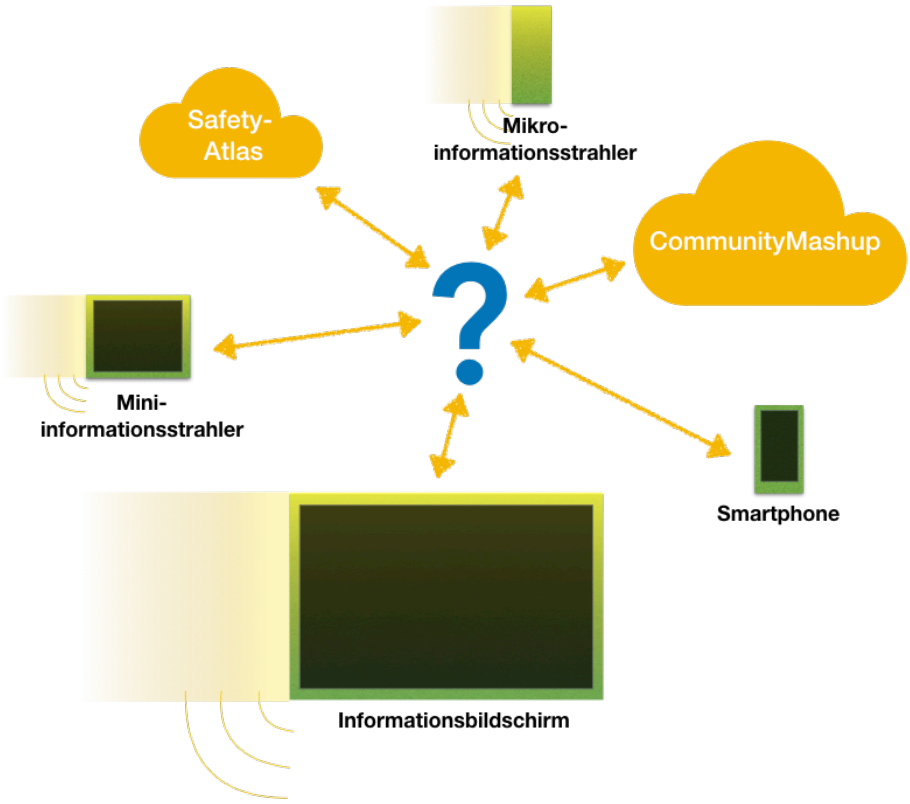

Abbildung 6 - Geräte- und Servicekomponenten in UL+

3.5.2. Schnittstellen & Informationsaustausch

Betrachtet man die Schnittstellen zwischen den Komponenten aus Abbildung 6 auf der Abstraktionsebene des Informationsaustauschs, so werden manche Beziehungen schnell klar – andere nicht. Dass alle Informationsstrahler bspw. mit dem CommunityMashup bzw. mit der Cloud kommunizieren müssen, um Stammdaten zu Orten und Organisationen zu abzufragen, fällt schnell auf. Doch das Anwendungsszenario erfordert auch komplexe Zusammenhänge zwischen den Strahlern, etwa dann, wenn Strahler als gemeinsame Einheit einen Nutzer zum Ziel führen sollen. Für solche Fälle muss das Konzept eine flexible, zuverlässige und erweiterbare Struktur aufweisen, die es ermöglicht, Informationen von der richtigen Quelle an das richtige Ziel zu übermitteln. Zu welchem Grad dieser Informationsaustausch über das CommunityMashup stattfindet, ist zu erarbeiten. Es ist auch eine direkte Kommunikation zwischen Strahlern denkbar.

Dem Informationsaustausch zugrunde liegt die Information. Zu identifizieren ist, welche Informationen wann, wie, von wo und wohin übertragen werden müssen, um der Anwendung gerecht zu werden. Daraus sind Informationsklassen zu abstrahieren, um Protokolle zwischen den Komponenten zu entwickeln. Das Konzept soll demnach auch aufzeigen, welche Datenobjekte wie übertragen werden.

3.5.3. Persistenz

Die Mikrostrahler haben vergleichsweise geringe Speicher- und Rechenkapazitäten. Anwendungskritische Abhängigkeiten sollten nur zu Komponenten mit gewissermaßen zuverlässigem und leistungsfähigem Charakter bestehen, etwa der Cloud. Persistenz im Sinne von Datenzuständigkeit gegenüber anderen Komponenten sollen die Mikrostrahler deshalb nicht aufweisen. Teils ist eine Zustandsbehaftung aber auch notwendig, etwa für die Konfiguration der

Strahler. Die Persistenz umfasst jedoch keine Informationen über Personen, Orte oder Organisationen. Solche Daten sollen dynamisch und während der Laufzeit eingeholt werden, einen Geräteneustart aber nicht überdauern. Nicht als Persistenz zu verstehen ist hingegen das Cachen von Informationen. Informationsstrahler dürfen sich Daten zu Personen, Orten und Organisationen merken, wenn diese mit einem zuverlässigen Mechanismus aktuell gehalten und zu einem angemessenen, nahegelegenen Zeitpunkt automatisch gelöscht werden. Damit kann insbesondere auch Datenschutzbedenken entgegengewirkt werden.

3.5.4. Anwendungslogik

Wie bereits bzgl. Informationsaustausch angedeutet, ist im Konzept darzulegen, welche Informationen über die Cloud geleitet werden, und welche nicht. Gleiches gilt für sämtliche Anwendungslogik, die für die Fachlichkeit implementiert werden muss. Ob bspw. die Fachlogik für die Routenführung eines Nutzers auf einem Informationsstrahler, in der Cloud oder auf dem Smartphone des Nutzers ausgeführt wird, ist zu klären.

3.6. Technik

In technischer Hinsicht lassen sich einige Hardware- und Software-Anforderungen formulieren, die vom Konzept erfüllt werden müssen oder die im Konzept als gegeben betrachtet werden können.

3.6.1. Hardware

Die Hardwareausstattung der Mikroinformationsstrahler hat einigen funktionalen und nicht-funktionalen Ansprüchen zu genügen. Zu den Funktionalen lassen sich sämtliche Schnittstellen zählen. Die Hardware muss Datenschnittstellen für die Kommunikation mit den anderen Komponenten unterstützen. Die Näherung von Personen soll über die iBeacon-Technologie erkannt werden, deshalb hat die Hardware Bluetooth LE-fähig zu sein. Bluetooth LE ist Teil von Bluetooth 4.0.

Für die Kommunikation zwischen den anderen Komponenten soll das Internet Protocol (IP) die Basis bilden (siehe auch 3.7.2 Internetanbindung).

Zu den nicht-funktionalen Kriterien der Hardware gehört eine ausreichende Leistungsfähigkeit. Der Mini-Computer soll fähig sein, eine gängige Linux-Distribution auszuführen und darin ein weiteres Linux-Betriebssystem wie Debian oder Raspbian zu virtualisieren. Die Virtualisierungsmöglichkeit soll gegeben sein, um Software-Deployments auf unterschiedlichen Abstraktionsebenen gewährleisten zu können. Näheres wird in 5.3 über den Prototyp geklärt. Prozessor- und Speicherkapazitäten müssen ausreichen, um Java 8 samt Java Virtual Machine und der eigentlichen Anwendung auszuführen.

Für das Konzept als Voraussetzung zu betrachten sind Benutzerschnittstellen für Licht und Audio. Für die Lichtkomponente sollte eine mehrfarbige, steuerbare Leuchte zur Verfügung stehen, die idealerweise auch Richtungshinweise, wie Pfeile, visualisieren kann. Außerdem notwendig ist eine Lautsprechereinheit (bspw. über AUX-Klinke) und ein Mikrofon.

Der Raspberry Pi 3 Model B gilt als verbreitetes Entwicklerkit und bietet für diese Arbeit ausreichend Ressourcen. Er ist ausgestattet mit einer Quad Core CPU, 1 GB Arbeitsspeicher,

WLAN, Bluetooth LE, AUX-Stereo-Ausgang und etlichen weiteren Schnittstellen[20]. Mit dem Sense HAT Modul ist das Gerät um eine 8x8-Farb-LED-Matrix erweiterbar[21], mit der sich ein großes Farbspektrum und auch Symbole wie Pfeile oder Buchstaben darstellen lassen. Für die Audioausgabe kann ein externer Lautsprecher verwendet werden. Mit einer externen USB-Soundkarte lässt sich auch eine Mikrofonschnittstelle ergänzen. Für dieses Konzept soll das Gerät die Referenz bilden.

3.6.2. Software

Auch die zu entwickelnde MIS-Software hat einigen Kriterien zu genügen, dabei spielen insbesondere allgemein bekannte Praktiken der Softwareentwicklung eine wichtige Rolle, darunter: Kopplung und Kohäsion, Erweiterbarkeit, Fehlertoleranz und Dokumentation. Zyklische Abhängigkeiten sind zu vermeiden.

Gerade in einem Umfeld, in welchem viele kleine Geräte vernetzt sind, soll das System auch einen nachgiebigen Charakter haben, d.h. mit Komponentenausfällen und -fehlern umgehen können.

3.7. Infrastruktur

Für den tatsächlichen Betrieb der Mikroinformationsstrahler relevant sind auch infrastrukturelle Voraussetzungen. Darunter Anforderungen an die Energieversorgung und Internetanbindung.

3.7.1. Energieversorgung

Für das Konzept darf angenommen werden, dass an den Installationsorten der Mikroinformationsstrahler eine übliche Stromquelle von 230V Spannung vorhanden ist oder erschlossen werden kann. Der Annahme zugrunde liegt bspw., dass an Montageorten wie Straßenlaternen und teils Bushaltestellen bereits eine Energieversorgung vorhanden ist. Nicht undenkbar ist auch eine Energieversorgung durch Photovoltaikkomponenten. Für Testaufbauten des Prototyps ist die Verwendung von Steckdosen naheliegend. Für den Einsatz des Raspberry Pi kann ein übliches 5V-USB-Netzteil genutzt werden.

3.7.2. Internetanbindung

Das Konzept setzt eine IP-basierte Netzwerk- und Internetanbindung voraus. Eine eigene öffentliche IP-Adresse für jeden einzelnen Mikroinformationsstrahler ist dabei nicht notwendig, ein NAT-Subnetz ist ausreichend.

Es steht eine Vielzahl an Schnittstellen und Verfahren zur Verfügung, um die Internetanbindung der Hardware zu gewährleisten. Besonders bekannt sind LTE, UMTS, GSM, ZigBee, Wi-Fi, Bluetooth oder auch kabelgebundene Lösungen. Auch Kombinationen sind denkbar, etwa ein WLAN-Router, der ein Subnetz aufspannt und per LTE mit dem Internet verbunden ist. Die Verfahren haben unterschiedliche Vor- und Nachteile, die stark von Installationsort und anderen situativen Begebenheiten abhängen. Zugang zu einem TCP/IP-Netzwerk wird im Konzept vorausgesetzt. Es abstrahiert die physischen Schnittstellen weg, deshalb wird hier keine spezielle

[20] https://www.raspberrypi.org/products/raspberry-pi-3-model-b/ (letzter Abruf: 26.07.2017)

[21] https://www.raspberrypi.org/products/sense-hat/ (letzter Abruf: 26.07.2017)

Lösung festgelegt oder empfohlen. Für den Prototyp-Testbetrieb kann am einfachsten auf LAN und WLAN zurückgegriffen werden.

3.8. Datenschutz

Zentrale Maßgabe an das Konzept ist, dass sensitive, vertrauliche und personenbezogene Daten bestmöglich geschützt werden. Um solche Informationen zu schützen, sollen sie nur auf dem persönlichen Nutzerendgerät gespeichert werden. Eine zentrale Verarbeitung solcher Daten ist zu vermeiden.

Zu diesen Daten gehören insbesondere Bewegungsprofile, Nutzerstandorte, Navigationsziele und persönliche, körperliche Beeinträchtigungen. Die Verarbeitung oder Übertragung solcher Daten soll möglichst nur dort stattfinden, wo tatsächlich notwendig.

Dar diese Arbeit grundsätzlich eine verteilte Anwendung aus vielen Informationsstrahlern darstellt, ist es unvermeidbar, dass personenbezogene Daten auf Teilen von ihnen verarbeitet werden. In diesem Fall ist zu achten, dass solche Daten nicht dauerhaft gespeichert werden. Es sollte möglichst verfolgt werden, dass Daten nur auf jenen Strahlern verarbeitet werden, die für die Anwendung tatsächlich nötig sind. Dies kann z.B. durch einen kleinen geographischen Radius um den Nutzer eingegrenzt werden.

4. Konzept

Dieses Konzept ist aus vielfältigen und intensiven Überlegungen auf verschiedenen Ebenen von Anwendung bis Technik entstanden. Der in Kapitel 2 vorgestellte Stand der Wissenschaft sowie die in Kapitel 3 vorgestellten Anforderungen bilden hierfür die Basis. Dabei wird sich zunächst besonders mit den Interaktionsmöglichkeiten beschäftigt, die mit einfachen Licht- und Audio-Komponenten erzielt werden können. Anschließend wird das UL+ Ausgangsszenario genauer betrachtet und erarbeitet, wie und wo solch einfache Interaktionen dem Nutzer im Szenario weiteren Mehrwert bieten können. Daraus entsteht ein ergänzendes Szenario, das um den Einsatz von Mikroinformationsstrahlern erweitert ist. Anhand dessen lässt sich ein fachliches Konzept erarbeiten, von dem technische Anforderungen und Konzepte abgeleitet wurden, begonnen bei Schnittstellen und Architektur und endend bei Implementierungsplänen. Dieses Kapitel erklärt aufeinander aufbauend dieses Konzept für Mikroinformationsstrahler.

4.1. Interaktionsmöglichkeiten

In Abschnitt 2.1 wurden einige aufkommende und begrifflich gefestigte Benutzerschnittstellen betrachtet. Ziel dieser Arbeit ist jedoch, sich auf sehr einfache Schnittstellen zu beschränken. Grenzt man die Auswahl auf Licht- und Audioschnittstellen, sowie die Fähigkeiten der iBeacon-Technologie ein, so fallen einige der klassischen Bedienkonzepte heraus.

Die nachfolgend aufgeführten Interaktionsmöglichkeiten lassen sich von den in 2.1 und 2.2 gezeigten Konzepten der MCI inspirieren. Besonders das – leider nicht wissenschaftliche – Projekt Intellistreets (siehe 2.2.1.7) bot hier Ideen. Die Möglichkeiten entspringen auch maßgeblich der Kreativität des Autors. Durch heuristische Methoden wird versucht, sich in das Szenario und in das Persona hineinzuversetzen und aus den gegebenen Schnittstellen Interaktionsmöglichkeiten abzuleiten. Soweit möglich wurden hier auch die in 2.1.4 vorgestellten Designregeln angewandt.

4.1.1. Licht

Als Licht-Komponente gegeben ist ein Leuchtmittel, das in einer 8x8-LED-Matrix auch einfache Symbole darstellen kann. Abbildung 7 zeigt einen Raspberry Pi mit installiertem Sense HAT Modul[22], wie er auch im Prototyp zum Einsatz kommt. Die Leuchtkomponente ist ein reiner Ausgabekanal, dessen informative Fähigkeiten begrenzt sind, aber dennoch Potential aufweisen. Schon auf einem Display in Tablet-Größe lassen sich viele und vielfältige Informationen wiedergeben, etwa Karten,

Abbildung 7 - Raspberry Pi mit Sense HAT

[22] https://www.raspberrypi.org/products/sense-hat/ (letzter Abruf: 27.07.2017), Bildquelle: https://www.raspberrypi.org/app/uploads/2017/05/Sense-HAT-plugged-in-1-1383x1080.jpg (letzter Abruf: 11.08.2017)

Fahrpläne und Umgebungshinweise. Solche Möglichkeiten gibt es bei einfachen Lichtstrahlern nicht, doch auch simple, farbliche oder symbolische Hinweise können dem Nutzer einen erheblichen Mehrwert bieten, etwa, wenn es sich um einen rettenden, richtungsweisenden Pfeil handelt. Besonders in blinkenden oder pulsierenden Modi ist auch denkbar, solche Lichtkomponenten zu nutzen um die Neugier und Aufmerksamkeit von Nutzern zu wecken. Der Blick von Nutzern könnte gezielt auf Objekte wie Sehenswürdigkeiten, Parkbänke oder Geschäfte gelenkt werden. Spinnt man diesen Gedanken noch etwas weiter, so ist vorstellbar, mehrere Lichtstrahler zusammenarbeiten zu lassen, um Nutzer entlang von pulsierend-leuchtenden Pfaden zu führen. In einer besonders smarten Variante bekommt der Nutzer eine eigene Lichtfarbe zugeteilt und kann Pfaden dieser Farbe folgen.

Eine weitere Möglichkeit der Lichtstrahler sind bestätigende oder ermunternde Hinweise, z.B. durch positive Farbimpulse oder Smiley-Symbole. Solche Hinweise könnten bspw. genutzt werden, wenn der Nutzer auf seinem Pfad einen Lichtstrahler passiert. Dem Nutzer könnten so motivierende Sentiments vermittelt werden. Gleichermaßen sind auch Warnhinweise, z.B. Ausrufezeichen vor roter Farbe, angebracht, die dem Nutzer gezeigt werden, wenn er von seinem Pfad abgekommen ist. Das Sense HAT Modul erlaubt in beschränktem Umfang auch die Darstellung eines Lauftexts, worüber kurze Worte oder Nachrichten als zusätzliche Hinweise eingeblendet werden können. Im Zusammenhang mit Sprachausgaben erscheint diese Funktion jedoch obsolet, für schwerhörige oder taube Menschen allerdings weiter interessant.

Besonders lichtstarke Strahler ließen sich auch als Beleuchtungsmittel als Ergänzung oder im Ersatz zur Straßenbeleuchtung einsetzen. Auch farbliche, atmosphärische Akzente könnten gesetzt werden.

Die Ideen für einen solchen Lichtstrahler sind nicht an ein Matrix-LED-Licht gekoppelt. Es sind auch noch simplere Umsetzungen vorstellbar, etwa eine Ampel-artige Leuchte mit drei farbigen Lampen, die ebenfalls Signale vermitteln können. Der mögliche Funktionsumfang reduziert sich dabei jedoch entsprechend.

Insgesamt ist festzustellen, dass sich auch mit einfachen Licht-Komponenten viele Interaktionsmöglichkeiten erdenken lassen, wenn man auf das Einsatzszenario bedacht ist. Für generische Zwecke bieten Displays (im Sinne von visuellen Komponenten) vermutlich einen deutlich größeren Lösungsraum.

4.1.2. Audio

Wie bereits in 2.1.2.4 Auditive Schnittstellen erläutert, lässt sich bei auditiver Interaktion zwischen Sprach- und Nicht-Sprach-Schnittstellen unterscheiden. Außerdem hat man bei einer Audio-Komponente nicht nur über Audioausgabe, sondern auch -eingabe nachzudenken. Nachfolgend werden einige Interaktionsmöglichkeiten in diesen beiden Dimensionen aufgearbeitet.

4.1.2.1. Ausgabe

Zunächst werden Audioausgaben betrachtet. Nicht-Sprach-Ausgaben sind bspw. Signale, Geräusche und Musik. Ähnlich zum Lichtstrahler lässt sich auch mit akustischen Mitteln die Aufmerksamkeit von Personen erlangen. Damit ein Mensch solche Signale wahrnehmen kann, muss er sich entweder nahe dem Informationsstrahler befinden, oder die Audioquelle muss entsprechend lautstark wirken. In letzterem Fall wären durchaus Szenarien wie mit Lichtstrahlern

vorstellbar: Der Nutzer folgt einer Geräuschquelle, möglicherweise sogar einem Geräuschpfad aus mehreren Audiostrahlern. Noch kreativer gedacht: Der Nutzer könnte der Melodie seines Lieblingslieds folgen, um nach Hause geleitet zu werden.

Ob dies eine tatsächlich praktikable Anwendung ist, lässt sich bezweifeln, schließlich kann die Geräuschkulisse im städtischen Raum durchaus laut sein und Anwohner könnten sich gestört fühlen. Doch im Nahbereich eines Audiostrahlers können einfache Signale in angemessener Lautstärke den Nutzer durchaus unterstützen. Erreicht ein Nutzer einen Strahler, spielt dieser einen bestätigenden und motivierenden Signalton. Dem Nutzer wird vermittelt, dass er sich in einer sicheren Umgebung befindet und jederzeit Beistand erwarten kann. Ebenso lässt sich ein Nutzer aber auch warnen, bspw. wenn er seinen geplanten Pfad verlässt oder in eine Gegend gelangt, die nicht mit Mikroinformationsstrahlern erschlossen ist. Hier lassen sich mit Audio-Komponenten vielversprechende Feedback-Möglichkeiten implementieren, wie auch von (Hoggan & Brewster, 2012) erläutert.

Bezieht man Sprachausgaben in die Überlegungen mit ein, ergeben sich vielfältige weitere Möglichkeiten. Mit Hilfe von Sprache sind nicht nur einfache Signale möglich, sondern auch die Wiedergabe komplexer Informationen. Gelangt eine Person in die Nähe eines Audiostrahlers, könnte eine synthetische Stimme nützliche Hinweise geben. Schon die einfache Aussage „Hallo Margot, Sie befinden sich auf dem richtigen Weg“ könnte ein starkes Gefühl von Sicherheit vermitteln. Vorstellbar sind auch Informationen wie „Ihr Bus kommt in acht Minuten“, „Auf der anderen Straßenseite befindet sich eine Sitzbank im Schatten“ oder „Ihre Freundin Brigitte ist nur zwei Straßen entfernt, Sie treffen sie bald“. Der Fantasie sind hier vermutlich kaum Grenzen gesetzt. Besonders kontext- und situationsbezogene, intelligent erschlossene Hinweise, die dem Nutzer implizit geboten werden, könnten nützlich sein.

4.1.2.2. Eingabe

Was die Audioeingabe betrifft, so sind Nicht-Sprach-Eingaben eine eher unübliche Form. Eingaben aus Rasseln, Summen oder Zischen scheinen keine besonderen Vorteile zu bieten – insbesondere gegenüber Spracheingaben.

Durch die Unterstützung von Spracheingaben (neben Ausgaben) kann ein Audiostrahler zu einem mächtigen, sprachgesteuerten Werkzeug werden. Strahler können nicht mehr nur implizit Informationen wiedergeben, sondern Nutzer können aktiv und explizit Informationen anfordern oder Aktionen auslösen. Die Interaktion wird natürlich.

In einer vereinfachten Umsetzung, bei welcher der Audiostrahler lediglich die Worte „Ja“ und „Nein“ als Eingabe akzeptiert, könnte ein Nutzer auf die Frage „Möchten Sie nach Hause geleitet werden?“ entsprechend reagieren. Ruft eine Person wiederholt das Wort „Hilfe“, könnte ein entsprechender Notruf abgesetzt werden. „Wie lange hat die Bäckerei Müller noch geöffnet?“, „Lass Brigitte wissen, dass ich gleich bei ihr bin“ oder „Wie komme ich nach Hause?“ hingegen sind Fragen und Aufforderungen, mit denen ein Nutzer auf vielseitige Weise mit Audiostrahlern kommunizieren könnte. Dabei ist leicht erkennbar, dass sich aus einer Frage auch ganze Konversationen ergeben können. Wie der Nutzer nach Hause kommen könne, könnte mit der Gegenfrage „Möchten Sie noch etwas spazieren oder gleich den Bus nehmen?“ hinterfragt werden.

Sprach- und Konversationsschnittstellen erleben mit neuen Diensten wie Amazon Alexa und Google Assistant[23] (siehe 2.1.2.5) derzeit einen Aufschwung. Ihr Potential soll auch in dieses Konzept einfließen.

4.1.3. iBeacons

Die dritte wesentliche Benutzerschnittstelle im Bunde bildet die iBeacon-Technologie. Sie befähigt Entfernungen zu Bluetooth LE Geräten zu approximieren und ihre Identität zu ermitteln.

iBeacons könnten besonders die Nutzerlokalisierung ermöglichen und damit als impliziter Schalter wirken, wenn Nutzer in die Nähe von Mikroinformationsstrahlern gelangen. Das Wissen über die Nähe von Personen ermöglicht es Strahlern, Aktionen auszulösen. So z.B. Reaktionen über die Kanäle Licht und Audio.

Wie in 2.3.4 erläutert, besteht das Bluetooth LE Protokoll aus verschiedenen Rollen. Es eröffnet unterschiedliche Implementierungsvarianten. Dabei stellt sich insbesondere die Frage, welches Gerät als Central, und welches als Periphal auftritt.

4.1.4. Multimodalität & Barrierefreiheit

Betrachtet wurden bisher die Komponenten Licht, Audio und iBeacon im Einzelnen. Dabei konnten schon einige Interaktionsmöglichkeiten ausgemacht werden. Vereint man diese in einem multimodalen System, so sind nochmals neue Varianten vorstellbar.

Licht-Signale ließen sich um Audio-Signale ergänzen und verstärken, um noch mehr Beachtung zu finden. Sprachausgaben könnten durch Lichtimpulse oder Equalizer-artige Lichteffekte untermalt und betont werden. Licht- wie auch Audio-Komponenten könnten auf die Nähe von Nutzern reagieren und entsprechend ihre Helligkeit, Lautstärke und Mikrofonempfindlichkeit justieren.

Besonders für den Kontext von UL+ könnte die Kombination von Licht und Audio einige Bedürfnisse der Barrierefreiheit befriedigen. Die Fähigkeiten für visuelle und auditive Wahrnehmung können in höherem Alter bekanntlich abnehmen. Für eine schwerhörige Person könnte es daher von Vorteil sein, wenn sie auf eine Licht-Komponente zurückgreifen kann. Gleiches gilt für sehgeschädigte Personen und Audio-Komponenten. Die Einstellung von Helligkeit und Lautstärke ließe sich zusätzlich an der körperlichen Verfassung des Nutzers orientieren.

4.2. Erweitertes Szenario

Mit Hilfe der in 4.1 erarbeiteten Interaktionsmöglichkeiten wird nachfolgend das bisherige Szenario (siehe 3.2) aufgearbeitet und ergänzt. Dabei werden zunächst Schwachstellen des ursprünglichen Szenarios ermittelt und anschließend Lösungen auf Basis von Mikroinformationsstrahlern erarbeitet. Abschließend werden die Lösungen in einem erweiterten Konzept gezeigt.

4.2.1. Versorgungslücken

Beim Lesen des Szenarios ist ein Problem besonders auffällig: Informationsbildschirme können nicht überall installiert werden, dafür sind sie bisher zu teuer und groß, besonders gegenüber

[23] https://assistant.google.com/ (letzter Abruf: 28.07.2017)

kleinen, einfachen Geräten. Doch dadurch entstehen Versorgungslücken, in welchen Nutzer nicht auf die Unterstützung durch Strahler zurückgreifen können. Im Szenario ist dies bspw. bemerkbar, als Margot von der Sparkasse zum Hardterbroicher Markt geht. Ihr stehen unterwegs nicht genügend Informationsstrahler zur Verfügung, um ausreichend mit Umgebungsinformationen versorgt zu sein. Sie muss Passanten nach dem Weg fragen. Es lässt sich vermuten, dass Margot sich in dieser Situation nicht besonders sicher gefühlt hat.

Zusätzliche Mikroinformationsstrahler können diese Versorgungslücken schließen. Sie können als kleiner und leichter zu montieren angenommen werden, so kann man ein dichtes Netz aus ihnen bilden. Durch die natürliche Spracherkennung und -verarbeitung und die hohe Dichte an Mikrostrahlern kann Margot ständig fragen, wie sie zum Hardterbroicher Markt komme. Sie könnte dann eine Antwort einer synthetischen Stimme mit Anweisungen erhalten. Andere Anwendungen können das Fragen nach Orten, Organisationen, Hilfe und ÖPNV-Fahrplänen sein.

4.2.2. Orientierungslosigkeit

Die lückenhafte Versorgung durch Informationsstrahler verursacht für Margot auch ein konkreteres Problem: Auf ihrem Weg zum Markt hat sie sich unter Umständen längere Streckenabschnitte von einer Karte einzuprägen. Dennoch verlor sie die Orientierung.

Mikroinformationsstrahler können Margot helfen, an ihr Ziel zu gelangen. Dafür sind nicht nur sprachliche Weganweisungen ein Mittel, sondern auch die Signalisierungsfähigkeiten der Strahler. Als Margot sich an der Informationstafel im Seniorenwohnheim für den Hardterbroicher Markt als Ziel entscheidet, könnte bereits ein zusätzlicher Unterstützungsprozess initiiert werden. Das Ziel auf der Karte könnte sich in eine Farbe färben, etwa Blau, ebenso wie die eingezeichnete Route dorthin. Als Margot ihre Komfortzone hinter der Sparkasse verlässt, könnten die Mikrostrahler, die auf dem Weg zum Markt liegen, in blauer Farbe pulsierend leuchten. Erreicht Margot einen der Strahler, würde dieser hell aufleuchten und einen Bestätigungston spielen. Außerdem könnte ein Pfeilsymbol dargestellt werden, dem Margot zum Markt folgen kann.

4.2.3. Unsicherheit

Nicht zuletzt durch die Orientierungslosigkeit gelangt Margot in eine Situation, in der sie sich unsicher fühlen dürfte. Doch auch andere Faktoren können dazu beitragen. Nicht zu wissen, wo die nächste medizinische Versorgung ist, wo der nächste Taxistand oder die nächste Parkbank ist, wie weit es noch bis zum Ziel ist oder wie im Notfall jemand zu erreichen ist, könnten das Gefühl verstärken. Ungewohnte situative Begebenheiten, wie Dunkelheit oder Unwetter, sind möglicherweise auch innerhalb der Komfortzone unangenehm.

Durch ausgefeilte Datenerhebungen und Algorithmen könnten Informationsstrahler die Gemütslage der Nutzer approximieren. Die Anbindung an bspw. Wetterdienste könnte den Strahlern eine Abschätzung über die lokale Wettersituation liefern. Bei großer Hitze und starker Sonneneinstrahlung ist die Gefahr eines Hitzeschlags besonders hoch, entsprechend könnte die Sprachsteuerung Empfehlungen für den Verbleib im Schatten oder in Gebäuden aussprechen. Bei erwartetem starkem Schneefall oder Regen könnten angemessene Aktivitäten vorgeschlagen werden.

Auch aus dem Positionswissen über einen Nutzer lassen sich Empfehlungen ableiten. Ermittelt man die mittlere Geschwindigkeit und die typische Tagesreichweite eines Nutzers, lässt sich entsprechend reagieren, wenn die Geschwindigkeit zwischen zwei Informationsstrahlern

nachlässt und die maximale Reichweite langsam erreicht wird: „Margot, Sie waren heute schon viel unterwegs. An der nächsten Kreuzung befindet sich eine Parkbank zum Ausruhen. Soll ich für Sie nach einem entspannten Heimweg suchen?“

Neben externen Diensten könnten weitere Datenerhebungen auch über zusätzliche Sensoren an den Strahlern stattfinden. Helligkeits- und Regensensoren seien hier nur als Beispiele genannt.

Durch intelligente Algorithmen haben Strahler das Potential noch deutlich mehr Lösungen abzudecken – trotz eingeschränkter Interaktionsmöglichkeiten. Womöglich bieten erst richtig ‚smarte’ Informationsstrahler den informativen Mehrwert, den ihr Name suggeriert.

4.2.4. Auffälligkeit

Nach (Brignull & Rogers, 2003) können öffentliche, interaktive Bildschirme für manche Menschen sozial unangenehm zu benutzen sein. Möglicherweise trifft dies auch für die Interaktion mit Mikrostrahlern zu.

Doch Mikroinformationsstrahler können auch sehr versteckt agieren. Bilden Mikrostrahler einen farbig-leuchtenden Pfad für einen Nutzer, den dieser verfolgt, so ist die Interaktion zwischen beiden kaum offenbar. Es wird angenommen, dass Nutzer über die Funktionsweise und Absicht der Mikrostrahler aufgeklärt sind, so sind die Hinweise für Nutzer verständlich. Doch Außenstehende bemerken die Interaktion lediglich an den Licht- und Audiosignalen, wenn der Strahler passiert wird.

Diese implizite und recht versteckte Unterstützung des Nutzers könnte diesem angenehm sein, schließlich ist er weniger darauf angewiesen unmittelbar vor einem Bildschirm zu stehen und dessen Inhalte zu studieren. Das Gefühl von Unbeholfenheit könnte hier reduziert werden.

4.2.5. Mehrbenutzerfähigkeit

Bei großen Informationsbildschirmen kann eine zeitgleiche Mehrbenutzerfähigkeit bspw. durch Aufteilen der Bildschirminhalte erzielt werden. Auch bei Mikroinformationsstrahlern kann die Situation eintreffen, dass mehrere Personen gleichzeitig einen Strahler benutzen möchten. Wird in einem Zeitpunkt zwei Nutzern ein Pfad geleuchtet, so reicht bereits die Nähe beider, um einen Konflikt zu verursachen.

Was die Licht-Komponente betrifft, kann durch Vernetzung der Strahler einige Abstimmung stattfinden, etwa um nutzerspezifische Farben zu ermitteln. Geht es um Lichtsignale und -farben, wäre eine abwechselnde Ausstrahlung vorstellbar. Kreuzen etwa die leuchtenden Pfade zweier Nutzer ein und denselben Strahler, so könnten beide nutzerspezifische Farben abwechselnd aufleuchten. Symbole, wie Pfeile, können in der entsprechenden Leuchtsequenz dargestellt werden. Audiosignale können ähnlich behandelt oder sequentiell ausgegeben werden.

Abbildung 8 - Konflikt bei Sprachsteuerung

Um bei Sprachausgaben die Mehrbenutzerfähigkeit zu ermöglichen, muss erkenntlich gemacht werden, welchem Nutzer welche Ausgabe gilt. Durch persönliche Ansprachen wie „Hallo Margot“ könnte dies erzielt werden. Eine sequentielle Ausgabe wäre dann möglich. Spracheingaben bilden hier eine besondere Hürde, da nicht immer klar erkennbar ist, wer spricht, wenn sich zwei Personen nahe einem Mikrostrahler befinden. Evtl. sind hier Approximationen über die Nähe des Nutzers möglich, doch die falsche Erkennung könnte fatale Auswirkungen haben. Im Zweifelsfall kann die Spracheingabe dann nur noch einen beschränkten Funktionsumfang anbieten, der nicht vom konkreten Nutzer abhängt. Abbildung 8 zeigt ein konkretes Problembeispiel, bei welcher eine Spracheingabe einem falschen Nutzer zugeordnet wird. Schwierig zu lösen sein dürften auch Fälle, in welchen unbeteiligte Personen per Sprachbefehl Anweisungen im Namen eines Nutzers geben. Die Privatsphäre und Identität des Nutzers hat hierbei geschützt zu werden. Im Beispiel in der Abbildung hätte Margot ggf. die Möglichkeit, durch Folgen des Mikrostrahler-Lichtpfads herauszufinden, wo Manuel wohnt.

Mit Hilfe von persönlichen Kenndaten, wie Name, Geburtsdatum oder PIN, ließe sich die Identität von Personen ggf. abfragen und prüfen, doch würde das eine zusätzliche Komplexität in die Interaktion bringen. Automatische Stimmerkennung, wie 1997 von Joseph Campbell vorgestellt, könnte eine einfache und integrierte Lösung sein (Campbell, 1997). Einen praktikablen Authentisierungs- und Authentifizierungsmechanismus für die Sprachsteuerung zu finden, würde den Rahmen dieser Arbeit sprengen. Dieses Konzept konzentriert sich auf die Anforderungen und Möglichkeiten der Interaktion.

4.2.6. Szenario 2.0

Unter dem Begriff „Szenario 2.0“ wird nachfolgend das Ausgangsszenario (sozusagen „Szenario 1.0“) erweitert und um einige der neuen Lösungen ergänzt. Dieses Szenario ist als Vision zu verstehen. Es zeigt Möglichkeiten intelligenter, kleiner Informationsstrahler im urbanen Raum auf. Auf Basis von Szenario 2.0 werden Systemarchitektur und Technik ausgearbeitet.

Das folgende Stück erweitert den Abschnitt des Ausgangsszenarios, in dem Margot von der Sparkasse aufbricht und sich auf den Weg zum Hardterbroicher Markt macht.

> Margot befindet sich am interaktiven Informationsbildschirm in der Sparkasse und hat sich den Hardterbroicher Markt als Ziel gesetzt, sowie die Herausforderung, heute ein Café zu besuchen, angenommen. Die Route und die Informationen zum Hardterbroicher Markt werden in blauer Farbe hervorgehoben.

Margot macht sich auf den Weg zum Markt und verlässt ihre Komfortzone. Sie sieht entlang des Weges mehrere kleine Geräte mit transluzenter Oberfläche an Straßenlaternen angebracht, die in blauer Farbe pulsierend leuchten. Sie erinnert sich an die blaue Färbung der Informationen zum Hardterbroicher Markt und folgt dem Lichtpfad. Auf Grund von Margots Sehschwäche, leuchten die Strahler für sie besonders kräftig. Als Sie sich einem der Mikrostrahler nähert, koppelt sich Ihr Smartphone automatisch mit dem Strahler. Der Strahler sendet einen Lichtimpuls aus und spielt einen Signalton. Ein kleiner leuchtender Pfeil weist ihr zusätzlich die Richtung. Margot fühlt sich bestätigt, sie ist auf dem richtigen Weg. Kurz danach erlischt das Licht des passierten Strahlers.

Als Margot am fünften Mikrostrahler angelangt, erkennt das Gerät, dass sie für den letzten Streckenabschnitt etwas länger gebraucht hat. Auf Basis von Margots persönlichen Profil und Umgebungsvariablen trifft das Gerät eine Entscheidung. Der Strahler sendet wieder einen Licht- und Tonimpuls. Dieses Mal ertönt danach jedoch eine Stimme: „Guten Tag Margot! Heute ist das Wetter sehr warm. Auf der anderen Straßenseite befindet sich eine Parkbank im Schatten. Möchten Sie diese benutzen?“ Margot nickt und sagt: „Gerne.“ Ein weiterer Licht- und Tonimpuls vermittelt Margot, dass das Gerät verstanden hat. Sie sucht nach einer Möglichkeit die Straße zu überqueren.

Auf der anderen Straßenseite, an der Parkbank, ist ein weiterer Mikrostrahler montiert. Er hat sich soeben in ein pulsierend-leuchtendes Blau gefärbt. Es folgt eine Stimme: „Guten Tag. Gleich wird eine ältere Dame diesen Platz aufsuchen. Bitte ermöglichen Sie ihr eine Sitzgelegenheit.“ Als Margot die Bank erreicht, ist diese frei und sie ruht sich einen Moment aus. Der Strahler bestätigt ihre Ankunft.

Nach ein paar Minuten rührt sich der Mikroinformationsstrahler erneut per Licht und Ton, es folgt: „Frau Nowak, bis zum Hardterbroicher Markt ist es nicht mehr weit. Das Café Borrelli ist dort in der Nähe und hat heute ein besonderes Angebot für Sie. Möchten Sie es wahrnehmen?“ Margot spricht: „Ja, in fünf Minuten gehe ich los.“ Das Gerät signalisiert eine Bestätigung.

Als sie sich wieder auf den Weg macht, wird sie weiter von den Mikroinformationsstrahlern begleitet. Margot gelangt am Markt an. Bei einem nahegelegenen Strahler fragt sie „Wo finde ich Brigitte?“, woraufhin ihr geantwortet wird „Sie war zuletzt am Informationsbildschirm in der Nähe des Haupteingangs des Marktes.“ Nach kurzer Suche finden sich beide.

Nachdem Margot und Brigitte sich gemeinsam den Markt angesehen haben, beschließen sie, noch etwas gemeinsam zu unternehmen. An einem Informationsbildschirm erkundigen sie sich nach Cafés in der Nähe. Sie finden ein Café, das heute Donauwelle für NEW-Kunden im Angebot hat. Sowohl Margot, als auch Brigitte bestätigen das neue Ziel. Die Zielinformationen werden Margot wieder in Blau dargestellt, in Brigittes Bildschirmbereich jedoch in Rot.

Beide machen sich auf den Weg und können sich wieder auf den Pfad verlassen, den die vorausliegenden Mikrostrahler Ihnen leuchten und weisen. Dieses Mal pulsieren sie jedoch zweifarbig-abwechselnd - rot und blau. Im Café verbringen beide einen angenehmen späten Nachmittag. Margot erfüllt die Herausforderung in ein Café zu gehen.

Als Margot sich auf den Heimweg macht, wird es bereits dunkel und Wolken ziehen über ihr zusammen. Bei einem Mikroinformationsstrahler erkundigt sich Margot, wie sie am besten nach Hause kommt. Da die Wetterdienste ein Gewitter prognostizieren, antwortet der Strahler: „Es wird schon dunkel und ein Unwetter ist angesagt. Möchten Sie, dass ich Ihnen ein Taxi rufe?“ Margot antwortet: „Das ist eine gute Idee.“ Automatisch veranlasst der Mikrostrahler die Bestellung eines Taxis und bestätigt dies gegenüber Margot: „Ihr Taxi ist in fünf Minuten da.“ Der Strahler pulsiert nun in gelber Farbe, um dem Taxifahrer und Margot den Standort zu signalisieren.

Margot gelangt sicher und trocken zu Hause an. Wäre sie zu Fuß gegangen und hätte den Bus genommen, wäre sie vermutlich ins Gewitter geraten.

Szenario 2.0 verdeutlicht, welchen Mehrwert der Einsatz von Mikroinformationsstrahlern als Ergänzung zu digitalen Informationstafeln bieten kann. Erkennbar ist jedoch auch bereits, dass hier Konflikte zwischen Szenario und Schutz der Privatsphäre entstehen. Die Antwort auf die Frage, wo Brigitte sei, ist bedenklich, denn nicht nur ist unklar, ob Margot von Brigitte konsequent berechtigt ist, ihren Standort zu erfahren, sondern auch erfordert es, dass Brigittes Standort über einige Ecken übertragen und verarbeitet werden muss.

4.3. Architektur

Szenario 2.0 hält einige Anforderungen an die System- und Softwarearchitektur bereit. Es ist festzuhalten, dass die Vision des Szenarios nicht vollständig in der Architektur umgesetzt wird. Im nachfolgenden werden die Ansprüche vom Szenario an die Architektur etwas eingeschränkt und daraus ein Architekturkonzept hergeleitet, dass diesen Ansprüchen standhält und aktuelle technische Ansätze verwirklicht.

4.3.1. Ausschließungen

Einige Teile des Szenario 2.0 sind nicht oder kaum mit dem Schutz von Nutzerdaten vereinbar. Ganz besonders Stellen, an welchen Standortinformationen über Margot und Brigitte ausgetauscht werden, sind bedenklich. Sie erfordern eine zentrale, über alle Nutzerstandorte aufgeklärte Komponente, die für solche Fragen konsultiert werden kann – etwa die Cloud. Dies widerspricht den Datenschutzanforderungen. Aus diesem Grund werden diesbezügliche Anwendungsszenarien für die Architektur nicht berücksichtigt.

Das Szenario setzt außerdem stark auf Sprachsteuerung. Über sie können Dienste des Mikrostrahlers in Anspruch genommen werden. Die Architektur behandelt diese Dienste auf einer höheren, abstrakten Ebene, auf welcher Spracheingaben als einfachere Eingaben verstanden werden können. Darunterliegende Ebenen des Natural Language Processing oder Natural Language Understanding werden hier nicht behandelt.

4.3.2. Grundlagen

Grundlegende Architekturfragen werden auf Basis der Dienste und Prozesse getroffen, die im Szenario 2.0 vorkommen. Dabei wird im Szenario ersichtlich, dass teils viele Informationsstrahler zusammenarbeiten müssen, um die Anforderungen zu erfüllen. Ein präsentes Beispiel ist die Routenführung anhand eines leuchtenden Pfades.

Anzumerken ist, dass im Folgenden mit den Begriffen Informationsstrahler oder Strahler auch die Makro- und Ministrahler, d.h. die digitalen Informationstafeln, gemeint sind. Sie

unterscheiden sich insofern voneinander, als dass sie verschiedene Interaktionsfähigkeiten haben, jedoch sind sie alle Bestandteile des Gesamtsystems – des Internet of Things.

Die serviceorientierte Architektur (kurz SOA) ist ein Architekturparadigma für (verteilte) Systeme, das Dienste bzw. Services als primäre Einheiten modelliert und diese in einem Gesamtsystem orchestriert. Services leiten sich häufig von den anwendungsbezogenen Prozessen ab und können auf unterschiedlichen Abstraktionsebenen angesiedelt sein. Ein höherer Service kann sich demnach aus mehreren niedrigen Services zusammensetzen. Dadurch sind u.a. eine hohe Wiederverwendbarkeit und Entkopplung der Komponenten möglich.

Oftmals bauen Dienste aufeinander auf oder es bestehen Abhängigkeiten zwischen ihnen, deshalb bedient es sich meist der synchronen Kommunikation. Des Weiteren ist beim Architekturentwurf und bei der Implementierung erforderlich, dass alle möglichen Dienste und zugehörige Schnittstellen bekannt sind.

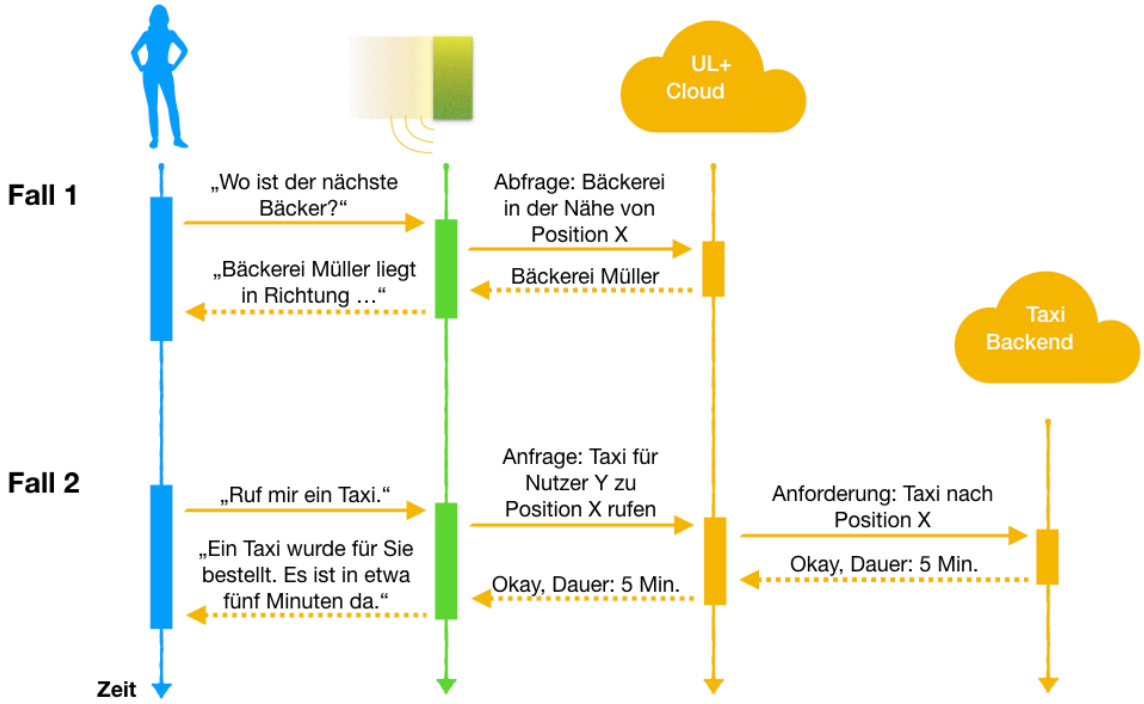

Abbildung 9 - Fallbeispiele 1 & 2 für SOA mit Cloud

Das CommunityMashup und der Safety-Atlas – also die Cloud – bieten sich als Beispiele für eine Mehrkomponenten-SOA an. Sie können klar definierte Dienste anbieten, die von Informationsstrahlern in Anspruch genommen werden. Abbildung 9 zeigt ein Sequenzdiagramm und zwei Fallbeispiele, bei denen mehrere Dienste miteinander verknüpft werden. Dabei ist die Abstraktionsebene noch relativ hoch. In Fall 1 kann für die Cloud für die Bearbeitung der Anfrage noch das Beanspruchen weiterer Services nötig sein, z.B. eine Abfrage bei einem Kartenservice und eine Abfrage bei einem Online-Branchenbuch.

Fall 2 zeigt die Integration eines weiteren Dienstes: dem Backend eines Taxiunternehmens. Alle Anfragen laufen synchron ab, die jeweiligen Prozesse blockieren. In beiden Fällen sind alle beteiligten Dienste, Komponenten und Zuständigkeiten wohl bekannt. Es gibt klare Schnittstellen und Prozesse, die konsultiert werden können.

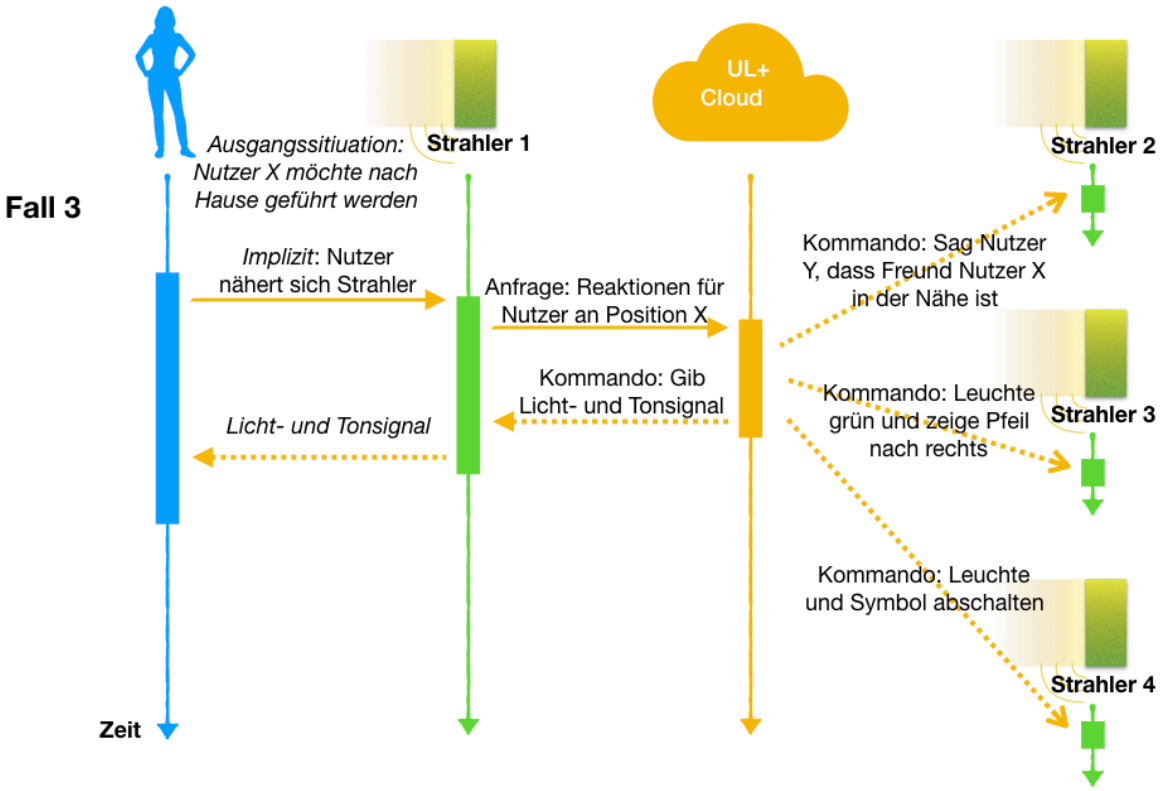

Abbildung 10 - Fallbeispiel 3 für SOA-Komplexität mit Cloud

Abbildung 10 zeigt ein Sequenzdiagramm für einen weiteren Anwendungsfall. Initial möchte der Nutzer nach Hause geleitet werden, diese Information ist dem System bereits bekannt. Beginn für die Sequenz ist dabei die Annäherung des Nutzers an einen Informationsstrahler (Strahler 1). Die Abbildung zeigt nur eine mögliche Lösung des Falls, auch die Granularität der Dienstinvokationen kann eine andere sein.

Strahler 1 konsultiert Dienste der Cloud, um Informationen über das weitere Vorgehen zu erhalten. Die Cloud beansprucht ggf. weitere Dienste (z.B. Kartendienst, Wetterdienst, Safety-Atlas, Taxidienst, etc.; nicht eingezeichnet). Auf Basis der Datenlage konsultiert sie weitere Strahler, um ihnen Anweisungen zu übermitteln. Der Einfachheit halber ist dies in der Abbildung mittels asynchroner Invokationen dargestellt. Dabei ist ersichtlich, dass die Kommandos sehr unterschiedliche Ausprägungen und Absichten verfolgen. Das situative Wissen, die Nutzerabsichten, die Fähigkeiten der Strahler, die Positionsinformationen, die Anwendungslogik und einiges mehr – sozusagen die vollständige Aufgeklärtheit – sind zentral in der Cloud verlangt, um entsprechende Kommandos zu erzeugen und abzusetzen. Dies scheint zu einer erheblichen Komplexität zu führen, auch wenn die Cloud selbst wiederum als Komposition vieler Dienste verstanden werden kann. Das Backend würde im Gesamtsystem eine umfangreiche Zuständigkeit erhalten.

Bei der Anfrage von Strahler 1 an die Cloud ist auch erkennbar, dass die konkrete Betitelung des Dienstes nicht trivial ist. Tatsächlich löst der Dienst viele Prozesse aus, doch erscheint es schwerfällig diese einem größeren, höheren Dienst zuzuordnen. Dass ein Nutzer über die Nähe eines Freundes informiert wird, ist in Fall 3 eher ein Seiteneffekt, den Nutzer X auslöst. Auch dass das Licht des unbenutzten Strahler 4 ausgeschalten wird, ist nicht direkt Teil des Dienstes „Führe Nutzer X nach Hause“.

Bemerkbar ist auch, dass es viele Strahler unterschiedlicher Typen (Mikro, Mini, Makro) und damit viele Dienstanbieter gibt, deren Dienstumfang heterogen ist. Die Heterogenität ist auch dadurch gegeben, dass die Dienste stark von ihrem situativen und lokalen Kontext abhängen. Der Dienst „Leuchte rot“ würde in diesem Beispiel von jedem Mikrostrahler angeboten werden. Ihr Ort und Kontext unterscheidet sie aber. Der Dienst „Frage Bäckereien in Mönchengladbach ab“ hingegen kann klar in der Cloud verortet werden.

Eine andere Architekturvariante könnte mit einem Polling-Prinzip realisiert werden, bei welchem alle Strahler regelmäßig Daten oder Kommandos aus der Cloud abfragen. Auch dies würde eine SOA verfolgen, doch bleibt hierbei eine hohe Systemkomplexität beim Backend. Außerdem könnte dies Nachteile für die Nutzerinteraktion bedeuten, da Feedback von Seiten des Strahlers ggf. verzögert kommt.

Die Anwendungsfälle sind sehr vielfältig, daher ist eine große Flexibilität und Erweiterbarkeit des Gesamtsystems wünschenswert. Besonders Fälle, an denen mehrere Informationsstrahler beteiligt sind, scheinen einem Paradigma von dynamischen Reaktionen zu folgen. Dabei ist eine konkrete Reaktion von vielen Faktoren abhängig, doch folgt sie meist auf eine vorangegangene Aktion.

Eine solche Aktion kann etwa ein Ereignis oder Event sein, ausgelöst bspw. durch einen Nutzer, durch einen anderen Strahler oder durch die Zeit. Je nach Situation und Kontext kann ein Event für einen Strahler relevant sei. Falls eine Relevanz vorliegt, kann er es verarbeiten und reagieren. Dabei fällt es leicht anzunehmen, dass die Reaktion im Sinne eines Events wieder andere Prozesse auslösen kann.

Die EDA hat gegenüber einer SOA einige Vorteile. Sie basiert bspw. auf loser, asynchroner Kommunikation, so sind Eventquellen und -empfänger gänzlich entkoppelt. Ein Event kann sich an mehrere Empfänger richten. Im Gegensatz zur SOA offeriert eine EDA keine wohldefinierten Funktionen und Services an Komponenten. Events werden den Komponenten zur Verfügung gestellt und diese sind frei darin sie zu verarbeiten und neue Events abzusetzen. Im Kontext dieser Arbeit veröffentlichen und konsumieren vor allem Strahler Events, nicht zuletzt, da sie die Ein- und Ausgabeschnittstellen für Nutzer bilden.

Als Mechanismus dafür, welche Komponente welche Events empfängt, kann das Publish/Subscribe-Modell benutzt werden. Dabei abonnieren Komponenten Themen und werden benachrichtigt, wenn eine Nachricht zu dem Thema veröffentlicht wurde.

Wie in 2.3 erklärt, sieht die konventionelle Umsetzung des Internet of Things vor, dass ‚Things’ im Sinne von Sensoren und Aktoren Daten an eine Cloud senden und von dort empfangen. Sind Städte mit tausenden von solchen Geräten ausgestattet, so kann dies erhebliche Datenmengen verursachen, die von der Netzwerkinfrastruktur getragen werden müssen. Auch für Echtzeitanwendungen bietet die Cloud nicht die richtige Plattform (Bonomi et al., 2012). Gleichzeitig werden auch kleine Geräte immer leistungsstärker. Aus diesen Gründen geht diese Arbeit einem neueren und an Fog Computing orientierten Ansatz nach, in welchem die Strahler selbst zu verarbeitenden Komponenten werden und nicht nur Sensoren und Aktoren einer Anwendung darstellen, die eigentlich in der Cloud läuft. Die Dezentralisierung reduziert die Belastung der Netzwerke und schafft geringe Latenzen.

Im Cloud-zentrischen Ansatz würde EDA die Hauptverarbeitung von Ereignissen weiterhin in der Cloud vorsehen. IS reagieren lediglich auf Events, die für sie als Befehle vorgesehen sind

oder erzeugen Events, die ein Nutzer verursacht. In der Fog-/Edge-peripheren Lösung findet ein Teil der Verarbeitung in den Endgeräten, also etwa in den Strahlern selbst, statt. Vom Nutzer verursachte Ereignisse können teils selbst verarbeitet werden, werden aber auch an andere Strahler weitergeleitet, damit diese darauf reagieren können.

Um der UL+-Anwendung gerecht zu werden, scheint für die Architektur der Mikroinformationsstrahler eine Kombination von SOA und EDA sinnvoll. Für die Kommunikation mit Backend-Instanzen wie dem CommunityMashup lässt sich mit SOA auf Services und Ressourcen zurückgreifen. Strahler wiederum können in einer EDA selbst auf Ereignisse reagieren, Ereignisse für andere Strahler erzeugen und die Services der Cloud in Anspruch nehmen.

4.3.3. Datenhoheit

Wie unter 3.1.3 erläutert, umfasst das CommunityMashup primär Stammdaten, darunter grundlegende Informationen zu Personen, Orten und Organisationen. Außerdem sind dort Aktivitäten, Veranstaltungen und Herausforderungen persistiert. Jegliche Informationsobjekte können miteinander verknüpft sein, so z.B. eine Person mit einer Organisation, eine Person mit einer anderen Person oder eine Person mit einer Herausforderung.

Aus Datenschutzgründen sollen sehr persönliche und vertrauliche Daten nicht zentral gespeichert werden. Deshalb sind Informationen wie körperliche Beeinträchtigungen, Bewegungsprofile, private Standorte und Navigationsziele/-routen auf dem persönlichen Nutzerendgerät – dem Smartphone – gesichert.

Die Informationsstrahler haben keine explizite Datenzuständigkeit, sie halten keine Daten dauerhaft. Davon ausgenommen ist ihre Konfiguration. Um korrekt arbeiten zu können, benötigen sie ein festes, initiales Wissen. Dazu gehören z.B. eine individuelle Geräte-ID, WLAN-Zugangsdaten, die eigenen GPS-Koordinaten und die eigene Himmelsausrichtung, Hinweise zu Art und Weise der Installation, sowie die IDs von benachbarten Informationsstrahlern. Zu ihrer Laufzeit nehmen Strahler Events entgegen und verarbeiten diese. Solche Events können über die Interaktion mit Nutzern (Sprachbefehl, Annährung) entstehen. Die Inhalte von solchen Nachrichten können im Rahmen der Verarbeitung auch in der Laufzeit erhalten bleiben, doch soll es keine Zuständigkeit gegenüber anderen bzgl. dieser Informationen geben. Events haben einen eher flüchtigen Charakter.

4.3.4. Protokolle

Die vorgestellte Architektur und das Wissen über die Datenhoheiten helfen dabei, passende Protokolle für die technischen Maschine-zu-Maschine-Schnittstellen (kurz M2M) auszuwählen.

In der Client-Server-Kommunikation hat sich REST mittlerweile einen Namen gemacht. Über eine REST-API können Clients mit den Ressourcen eines Servers arbeiten. Meist umgesetzt mit HTTP, werden Ressourcen dabei mit URIs identifiziert und Operationen mit HTTP-Verben beschrieben. Die öffentliche Schnittstelle des CommunityMashups offeriert eine REST-API, die CRUD-Operationen auf der Datenbasis erlaubt. Die Datenserialisierung findet dabei von und nach XML statt. Die IS nutzen diese Schnittstelle für ihre Belange.

Für die Implementierung einer EDA wird der Einsatz eines Publish/Subscribe-Modells präferiert. Während es viele Protokolle dafür gibt, qualifiziert sich MQTT durch seine Schlankheit besonders für IoT. Es soll daher für die Verteilung von Events genutzt werden. Auf

Serialisierungsebene kommt JSON als leicht lesbares, aber im Vergleich zu XML etwas kompakteres Format zur Anwendung.

4.3.5. Smartphone-Anbindung

Die Anbindung der Nutzer-Smartphones stellt eine besondere Herausforderung dar, da hier zusätzlich das iBeacon-Protokoll zum Einsatz kommt. Wie in 2.3.4 erklärt, ist das Protokoll rein unidirektional und überträgt nur sehr wenige Informationen. Grundsätzlich hätten sowohl IS, als auch Smartphone die Fähigkeit, als iBeacon-Transmitter zu agieren, also eine Kennung auszustrahlen. Hier ist zu überlegen, welche der beiden Komponenten als Transmitter, und welche als Empfänger realisiert wird.

Schlüssig erscheint, dass der IS als Sensor für die Annäherung fungiert und die Annäherung als Eingabe aufnimmt, um sie zu verarbeiten. In diesem Fall ist das Smartphone der Transmitter. Dabei ergibt sich das Problem, dass das Smartphone wichtige Nutzerdaten hält, die zur Verarbeitung an den IS übermittelt werden müssen. iBeacons eignen sich mit ihren insgesamt 4 Byte (Major und Minor) Übertragungskapazität nicht, um dies flexibel und erweiterbar zu implementieren.

Dienen IS als Transmitter, so können Smartphones ihre Nähe zu Strahlern erkennen. Da die iBeacon-Kommunikation unidirektional ist, ist aber auch hier kein Datenaustausch möglich. Denkbar wäre, dass in diesem Fall automatisch eine Bluetooth LE Kopplung stattfindet und darüber die notwenigen Informationen ausgetauscht werden. Da dieses Konzept jedoch schon einen Austausch- und Event-Bus in Form von MQTT nutzt und davon ausgegangen werden kann, dass Smartphones ans Internet angebunden sind, bietet sich an, diesen Kanal für die Nutzerdatenübermittlung einzusetzen.

Ein IS nimmt also die Rolle eines iBeacon-Transmitters ein und Smartphones als Empfänger erkennen Annäherungen an diesen IS. Das Smartphone verbindet sich beim iBeacon-Empfang mit einem MQTT-Broker und veröffentlicht ein Event über eine Nutzerannäherung, das auch (persönliche) Informationen über den Nutzer enthält. Das Topic der Veröffentlichung enthält dabei eine IS-Kennung, die dem Major/Minor des Transmitters entspricht. Der IS wiederum abonniert dieses Topic und bekommt vom MQTT-Broker das Event des Smartphones zugestellt.

Festzuhalten ist, dass das Smartphone bei Annäherung an einen Strahler nur einmal ein MQTT-Event absetzen darf. Da iBeacon-Transmitter frequentiert ausstrahlern, darf nicht jede einzelne Erkennung in einem Event resultieren. Auf Smartphone-Seite hat hier ein geeigneter Mechanismus implementiert zu werden, der ein mehrfaches Absetzen einer solchen Nachricht verhindert. Durchaus denkbar und für verschiedene Szenarien nützlich ist neben einem Annäherungsevent auch ein Entfernungsevent, das ausgelöst wird, wenn sich ein Nutzer von einem Strahler wieder entfernt.

Ebenfalls auf Seite des Smartphones hat implementiert zu werden, bei welcher genauen Entfernung ein Annäherungsevent abgeschickt werden soll. Je nach Signalstärke des Transmitters kann ein iBeacon schon in mehreren Dutzend Metern Entfernung erkannt werden. Über das iBeacon-Protokoll kann eine Entfernung in Metern approximiert werden. Das Konzept nimmt an, dass das Smartphone ein Annäherungsevent dann sendet, wenn sich der Nutzer in etwa ein bis zwei Metern Entfernung zu einem IS befindet.

4.3.6. Event-Reichweite

Mit der Event-Orientierung als Ansatz stellt sich die Frage, wie und an welche Komponenten Events verteilt werden. Im Konzept sind in den meisten Fällen nur in der Nähe befindliche IS von Nutzerinteraktion betroffen, sprich jeweils etwa zwei bis drei Strahler in jeder Richtung. Aus diesem Grund soll – wenn nicht anders nötig – die Veröffentlichung und Verteilung von Events lediglich einen nahen geografischen Raum um den Nutzer umfassen.

Um dies zu erzielen kann der Publish/Subscribe-Mechanismus von MQTT genutzt werden. Strahler veröffentlichen dabei Events, die sie selbst erzeugen, auf ein direkt ihnen zugeordnetes Topic. Das Topic enthält demnach eine eindeutige Strahler-ID. Unmittelbar benachbarte Strahler

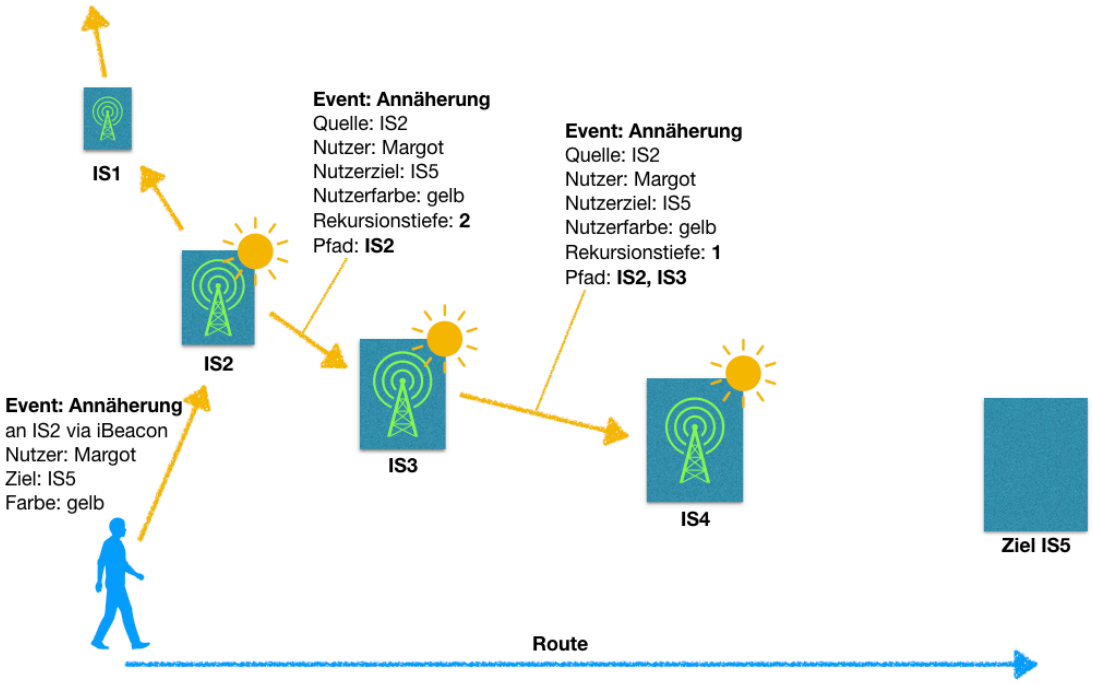

Abbildung 11 - Event-Propagation am Beispiel einer Benutzerroutenführung

sind mit dem Wissen über ihre Nachbarn konfiguriert, abonnieren dieses Topic und werden über die Events ihres Nachbarn benachrichtigt. Dadurch können auch andere IS auf die Annäherung eines Nutzers reagieren.

Möchte man eine noch größere Reichweite erzielen, so lässt sich dies mit einer Art Weiterleitungsmechanismus erzielen. Strahler veröffentlichen eingehende Events dann wiederum auf ihrem eignen Topic und stellen sie damit ihren Nachbarn zur Verfügung. Diese Propagation ist mit einem rekursiven Ablauf vergleichbar, der immer tiefer zu anderen Strahlern vordringt. Um dies zu beschränken, kann eine Rekursionstiefe festgelegt und mit übertragen werden. Bei jeder weiteren Veröffentlichung wird dieser Wert dekrementiert, bis er bei null angelangt ist und der Prozess gestoppt ist.

Abbildung 11 zeigt die Propagation von Events über Informationsstrahler anhand des Beispiels einer Nutzerroutenführung. Die dem Nutzer vorausliegenden Strahler sollen in einer einheitlichen Farbe leuchten, um den Nutzer an sein Ziel (hier IS5) zu führen. Zur Vereinfachung ist in der Abbildung kein MQTT-Broker dargestellt. Die Pfeile zeigen, welche Events von welcher Quelle zu welchem Empfänger gelangen bzw. wo sie weiterverarbeitet werden.

Der Prozess wird mit der Annäherung des Nutzers an IS2 initiiert. IS2 erhält dieses Event, wie in 4.3.5 beschrieben, via MQTT vom Smartphone. Dieses verarbeitet er, erzeugt aber auch ein Propagationsevent mit einer Rekursionstiefe von 2 und setzt es auf sein eigenes Topic ab. IS1 und IS3 haben das MQTT-Topic ihres Nachbarn abonniert und empfangen daher ebenfalls das Annäherungsevent. Sie verarbeiten es hinsichtlich des Nutzerziels. IS1 kommt zu dem Schluss, nicht auf der Route des Nutzers zu liegen, IS3 hingegen schon, deshalb beginnt er zu leuchten. Beide propagieren das Event weiter, jedoch mit einer um 1 dekrementierten Rekursionstiefe. Auch IS4 empfängt das Event und reagiert durch Licht. Er veröffentlicht das Event jedoch nicht weiter, da die festgelegte Rekursionstiefe erreicht wurde. Erst wenn sich der Nutzer IS3 nähert, wird die Tiefe von 2 ausreichen, um auch IS5 zu erleuchten.

Da IS2 auch das Topic von IS3, und IS3 das Topic von IS4 abonniert hat, könnte es zu Rückkopplungen/Zyklen kommen, bei denen IS2 (IS3) wiederum das von IS3 (IS4) empfangene Event veröffentlicht. Um dies zu verhindern ist in der Event-Nachricht ein Pfad-Parameter enthalten, der aufzeigt, welche IS von einem Event schon passiert wurden. Jeder IS erweitert es um seine eigene ID beim Veröffentlichen. Kommt das gleiche Event zurück, erkennt der IS daran, dass er das Event bereits verarbeitet hat und es verwerfen kann. Aus diesem Grund sind in der Abbildung auch keine rückwirkenden Pfeile eingezeichnet.

Für den realen Einsatz ist die Rekursionstiefe anpassbar. Im Fall der Routenführung dürfte es jedoch ausreichen, wenn die jeweils nächsten zwei bis drei Strahler leuchten, damit der Nutzer dem Pfad folgen kann.

Der durch diese Propagation implementierte Mechanismus ähnelt einer Rundfunkübertragung, die alle Empfänger innerhalb eines gewissen Radius erreicht. Obwohl der technische Mechanismus ein gänzlich anderer ist, wird der Mechanismus nachfolgend oft auch mit Broadcasting beschrieben.

4.3.7. Architekturbild

Architekturgrundlagen, Datenhoheiten und Protokolle erlauben nun ein Architekturbild zu zeichnen, welches die Ideen der Architektur wiederspiegelt. Abbildung 12 zeigt das Architekturkonzept auf der Abstraktionsebene der beteiligten Komponenten.

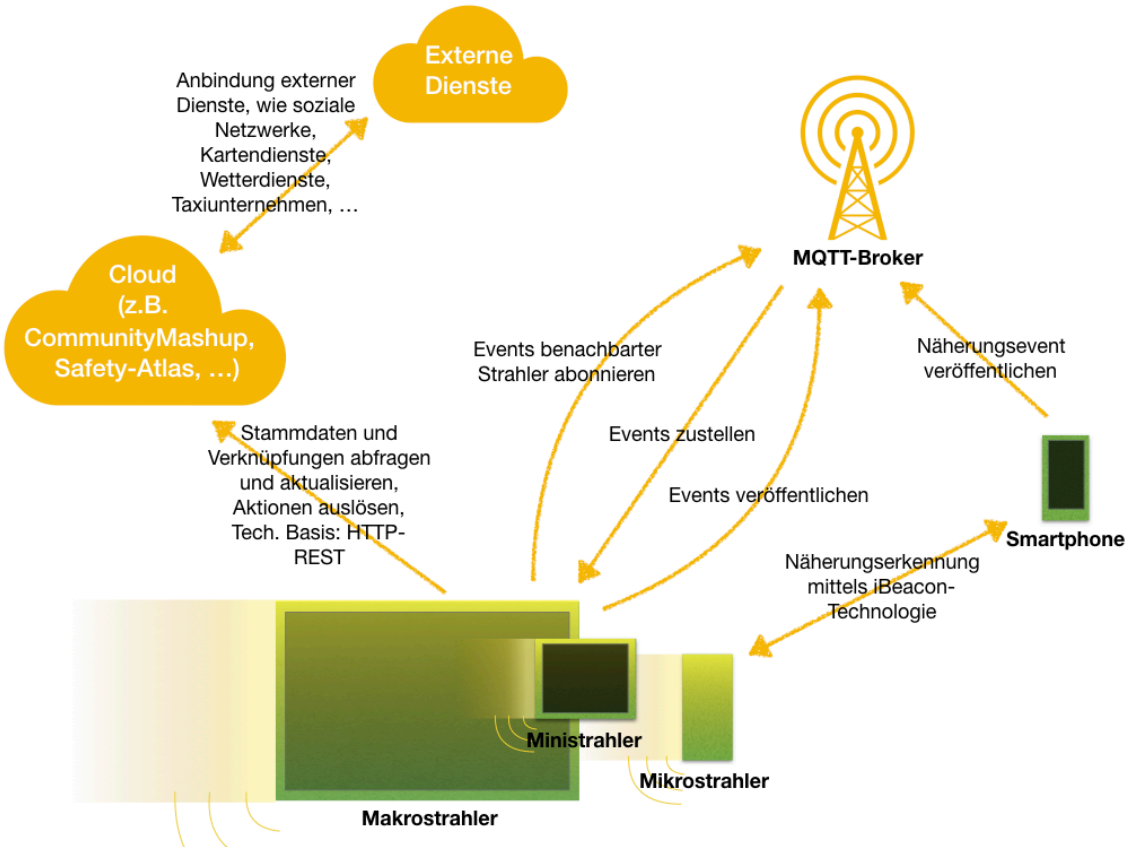

Abbildung 12 - Architekturbild auf Komponentenebene

Die IS spielen darin eine zentrale Rolle der Anwendungsdomäne, da sie die Schnittstelle zum Benutzer sind. Deshalb sind die IS insbesondere damit gekennzeichnet, dass sie Stammdaten und Zusammenhänge aus der Cloud anfordern oder darüber Dienste in Anspruch nehmen. Über die Cloud sind externe Dienste und ihre Funktionalität angebunden, um sie den IS zur Verfügung zu stellen. IS selbst abonnieren für sie relevante MQTT-Topics bei einem MQTT-Broker und veröffentlichen dort Events. Die Nutzerannäherung wird über die iBeacon-Technologie erfasst, aber über MQTT bekannt gemacht.

Der MQTT-Broker kommt als neue Komponente zum Gesamtbild hinzu. Er kann als Teil einer Middleware-Schicht betrachtet werden.

4.3.8. Szenario-Gegenprüfung

Das Szenario umfasst einige komplexe Anwendungen. Um die Umsetzung auf Basis der vorgestellten Architektur besser zu verstehen, wurden in der nachfolgenden Tabelle drei Use Cases aus Szenario 2.0 extrahiert und ermittelt, welche Komponenten an dem Vorgang beteiligt sind, sowie welche Zuständigkeiten ihnen zuzuordnen sind.

Tabelle 1 - Use Cases: Beteiligte Komponenten und Zuständigkeiten

#	Use Case	Beteiligte Komponenten	Zuständigkeiten
1	Margot nähert sich im Rahmen einer Routenführung einem Mikroinformationsstrahler **Vorbedingung** Erteilter Auftrag der Routenführung nach Hause; Route bereits berechnet und auf Smartphone gespeichert **Nachbedingung** Benachbarte Strahler auf Route leuchten farbig, passierter Strahler bestätigt und geht aus	Angenäherter Strahler (**ISa**) Benachbarte Strahler (**ISb**) Smartphone (**SP**) Externer Kartendienst (**KD**)	**Daten** SP: Route/Ziel **Erzeugte Events** SP: Annäherung (inkl. Route/Ziel) ISa: Annäherung (inkl. Route/Ziel) **Konsumierte Events** ISa: Annäherung (von SP) ISb: Annäherung (von ISa) **Verarbeitung** ISa: Reaktion auf SP-Event ISb: Reaktion auf ISa-Event (indirekt SP) **Benutzte Services** ISa, ISb: Ermittlung, ob IS-Standort auf Nutzerroute liegt (KD) **Ausgabe** ISa: Ankunft bestätigen, dann Licht abschalten ISb: Farbiges Licht einschalten
2	Margot nähert sich im Rahmen einer Routenführung einem Mikroinformationsstrahler und bekommt auf Grund heißer Wetterverhältnisse die Empfehlung eine nahegelegene Parkbank zu benutzen **Vorbedingung** Wie #1, zusätzlich heiße Wetterverhältnisse und benachbarter Strahler mit Sitzbank **Nachbedingung** Wie #1, zusätzlich Sprachausgabe über Sitzbankempfehlung und Sprachausgabe über Sitzbankfreimachung bei benachbartem Strahler	Wie #1, zusätzlich: Externer Wetterdienst (**WD**) Cloud (**C**) Benachbarter Strahler mit Sitzgelegenheit (**ISc**)	**Daten** SP: Route/Ziel WD: Lokaler Wetterstatus C: Orte (Parkbank) **Erzeugte Events** SP: Annäherung ISa: Annäherung, Sitzplatzwunsch **Konsumierte Events** Wie #1, zusätzlich ISc: Annäherung (von ISa), Sitzplatzwunsch (von ISa) **Verarbeitung** ISa: wie #1, zusätzlich Anwendung eines Sitzbedarfsalgorithmus ISb: wie #1 ISc: Reaktion auf Sitzplatzwunsch **Benutzte Services** wie #1, zusätzlich ISa: Anfrage Wetter (WD), Anfrage benachbarter Strahler mit Sitzgelegenheiten (C) **Ausgabe** Wie #1, zusätzlich

			ISa: Sprachausgabe ISc: Sprachausgabe
3	Margot nähert sich einem Mikroinformationsstrahler und spricht „Ruf mir ein Taxi.“ **Nachbedingung** Taxi beim örtlichen Taxiunternehmen an Margots Standort gerufen	Angenäherter Strahler (**ISa**) Smartphone (**SP**) Externer Taxidienst (**TD**)	**Daten** - **Erzeugte Events** SP: Annäherung ISa: Annäherung (hier irrelevant für Use Case) **Konsumierte Events** ISa: Annäherung (von SP) **Eingabe** ISa: Sprachbefehl zum Taxiruf **Verarbeitung** ISa: Reaktion auf SP-Event, Verarbeitung Sprachbefehl **Benutzte Services** ISa: Taxiruf (TD) **Ausgabe** ISa: Taxiruf bestätigen, ggf. Ankunftszeit nennen

Um die Zuständigkeiten der Komponenten besser zu zeigen, greifen die IS hier direkt auf externe Dienste zu – anders als in Abbildung 12 dargestellt.

4.4. Sicherheit

Um die Architektur bzw. das IS-System grundlegend abzusichern, werden einige Vorkehrungen getroffen, die hier kurz erläutert werden.

Um generelles Abhören der Kommunikation zwischen Komponenten zu verhindern, sollen jegliche Verbindungen verschlüsselt sein. Im Fall der HTTP-Kommunikation mit dem CommunityMashup oder der Cloud bietet sich TLS (HTTPS) an. Gleiches gilt für die Kommunikation zwischen MQTT-Clients und der MQTT-Broker-Middleware. Auch hier ist der Einsatz von TLS möglich.

Darüber hinaus soll der Zugriff auf Ressourcen beschränkt werden. Da in MQTT ohne weitere Spezifikation jeder Client jedes Topic abonnieren und auf jedes Topic beliebige Nachrichten veröffentlichen kann, soll der Zugriff auf Topics im Sinne von Ressourcen beschränkt werden. Ansonsten ist etwa denkbar, dass ein Angreifer die Aktivitäten eines Nutzers mitliest oder gefälschte Näherungsevents übermittelt. Sowohl IS, als auch Smartphones dürfen deshalb nur auf speziell für sie vorgesehene Topics veröffentlichen und nur für sie vorgesehene Topics abonnieren. Dazu mehr unter 5.4. MQTT bietet hier die Möglichkeit einer Benutzername-Passwort-Authentifikation, auf dessen Basis auch Ressourcen-Autorisierung möglich ist. Demnach erhält jeder Nutzer und jeder IS eine eigene MQTT-Kennung, die den Handlungsrahmen entsprechend wiedergibt.

Da die Architektur aus einer Vielzahl kleiner und eher leistungsschwacher Komponenten besteht, muss damit gerechnet werden, dass von Zeit zu Zeit auch einzelne IS ausfallen. Da die IS nach

dem aktuellen Architekturbild besonders durch den Propagationsmechanismus miteinander verbunden sind, braucht es eine Lösung, um mit fehlerhaften Strahlern umzugehen, bzw. um resilient über sie hinweg zu blicken. Im Sinne der Ausfallsicherheit ist es hier möglich, bei defekten Strahlern die benachbarten Strahler neu zu konfigurieren, um den fehlerhaften Nachbarn zu überspringen. Es wird dann nicht mehr das MQTT-Topic des Nachbarn, sondern das des übernächsten Nachbarn abonniert, bis der eigentliche Nachbar wieder funktionstüchtig ist. Kommt ein zentraler Konfigurationsserver zum Einsatz, lässt sich dieses Prozedere auch automatisieren, indem der Server regelmäßig den Gesundheitszustand der IS abfragt und im Problemfall dessen Nachbarn neu konfiguriert.

5. Prototyp

Im Rahmen dieser Arbeit wurde ein Prototyp nach Vorbild des Konzepts entwickelt. In ihm werden einige der Interaktionsmöglichkeiten und Architekturpläne umgesetzt. Der Prototyp dient rein illustrativen Zwecken und ist noch nicht für einen tatsächlichen Einsatz geeignet. Die Umsetzung des Konzepts in einen Prototyp zeigt aber, an welchen Stellen die Architekturentscheidungen Unzulänglichkeiten aufwiesen.

5.1. Funktionsumfang

Der Funktionsumfang des Prototyps gegenüber dem Konzept bzw. gegenüber Szenario 2.0 ist reduziert. Ziel ist zu zeigen, wie sich welche Funktionen mit dem Architekturkonzept umsetzen lassen. Um einen angemessenen Rahmen zu bilden, konzentriert sich der Prototyp auf die Interaktionsmöglichkeiten der Licht- und Symbolausgabe, der Sprachausgabe und der Annäherungserkennung. Spracheingabe ist auf Grund der Komplexität nicht Teil des Prototyps.

Um die Fähigkeiten des Modells zu zeigen, wurde der Prototyp um einige Funktionen ergänzt, die im Konzept nicht beschrieben wurden. Dies betrifft insbesondere Funktionen zum Ausführen von Simulationen, bspw. einer Nutzerannäherung.

Hauptziel des Prototyps ist die Fähigkeit, Nutzer entlang einer Route zu führen. Dabei spielen insbesondere die Licht- und Audioausgabeschnittstellen eine Rolle. Wie im Szenario beschrieben, soll eine Route im Sinne eines Strahler-Pfads mit Hilfe von farbigen Leuchten gezeigt werden. Bei unmittelbarer Annäherung an einen Strahler soll dieser ein Bestätigungssignal geben, sowie einen Richtungspfeil zeigen.

Darüber hinaus soll der Strahler einfache Sprachausgaben tätigen können, die implizit und auf eigene Initiative ausgelöst werden und auf Basis der Situation, etwa Wetterbegebenheiten oder körperlichen Beeinträchtigungen des Nutzers, beruhen.

Da einige der nicht in dieser Arbeit entwickelten Komponenten (CommunityMashup, Informationstafeln, etc.), nicht den für diesen Prototyp benötigten Entwicklungsstand haben, wird vom Konzept teilweise abgewichen. Bspw. ist an das CommunityMashup kein Wetterdienst angebunden, deshalb wird hier auf andere APIs ausgewichen.

5.2. Softwarearchitektur

Die Softwarearchitektur des Prototyps verfolgt Prinzipien objektorientierter Programmierung. Sie ähnelt in gewisser Hinsicht der Komponentenarchitektur, denn auch hier sind Konzepte von SOA und EDA wiederzufinden.

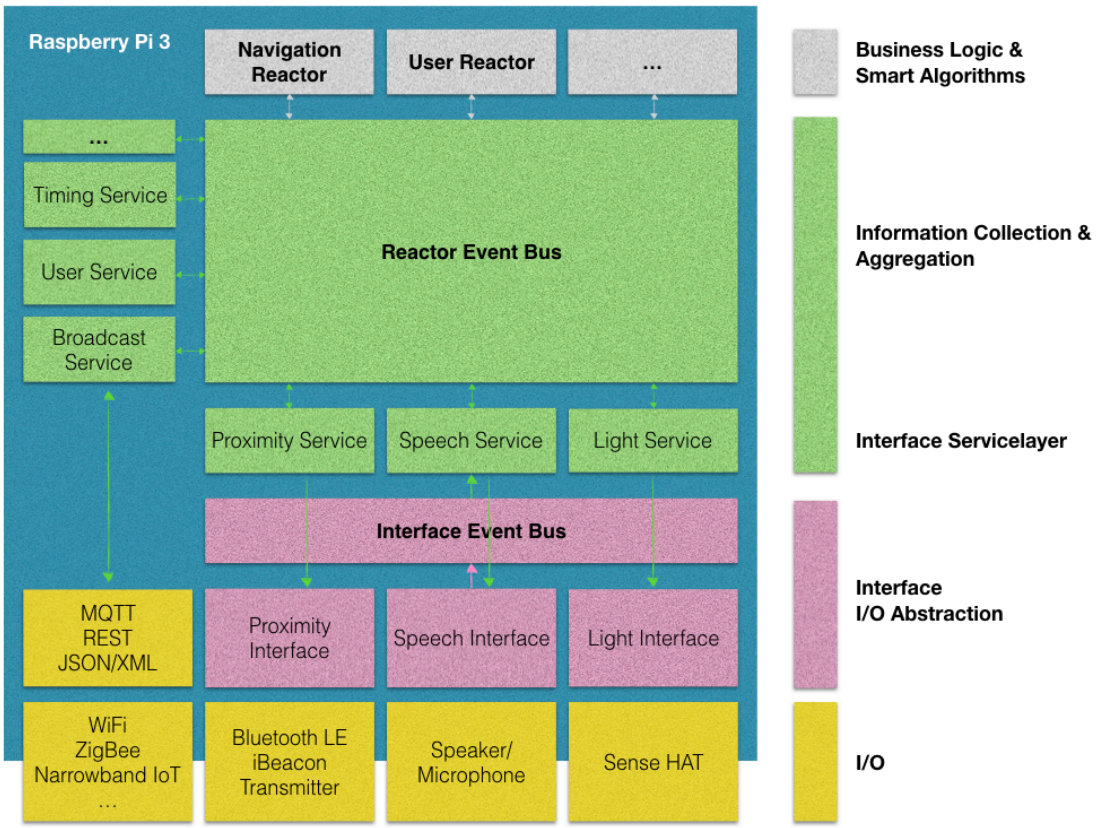

Abbildung 13 - Übersicht Softwarearchitektur

Abbildung 13 gibt einen Überblick über das interne Architekturkonzept. Unten gelb dargestellt ist eine Ebene externer Schnittstellen, etwa die Geräte der Benutzerschnittstellen, sowie die technischen Internet- und Bluetooth-Schnittstellen. Sie kennzeichnen hier den Übergang zum Äußeren der Mikrostrahler-Software.

Angebunden sind die Schnittstellen über hiergenannte Interfaces (lila). Sie abstrahieren die genaue Funktionsweise der Schnittstellen weg und ermöglichen eine einheitliche Zugriffsmöglichkeit. Für etwa Sense HAT steht eine zugehörige Bibliothek nur in Python zur Verfügung, so wird für ein Sense HAT Light Interface die benötigte Funktionalität in Python implementiert und über eine Konsolenschnittstelle darauf zugegriffen. Ähnliches gilt für den iBeacon-Transmitter. Das iBeacon Proximity Interface implementiert die Transmitter-Funktion, indem es via Konsole auf die BlueZ-Linuxanwendung für Bluetooth zugreift. Das Speech- oder Audio-Interface ist für die Implementierung der Lautsprecher- und Mikrofon-Nutzung gedacht.

Die Interfaces können gewissermaßen als Serviceanbieter für die oberhalb liegenden Interface Services (grün) betrachtet werden. Hier ist ein SOA-Muster erkennbar. Die Interface Service Schicht stellt eine semantisch höhere Ebene da. Während die Interface Schicht einfache Methoden auf den Ein- und Ausgabegeräten implementiert, wird auf Interface Service Ebene bereits ein Zusammenhang zur Anwendungsdomäne hergestellt. Interface Services können durch Methodeninvokation auf den Interfaces Ausgaben erzeugen. Werden an den Interfaces Eingaben erzeugt, so werden diese in den Interface Event Bus abgesetzt. Interface Services abonnieren diesen Bus und werden über die Eingaben benachrichtigt. Hier ist ein EDA-Muster erkennbar. Der Event Bus implementiert ein Publish/Subscribe-Modell.

Die Interface Services wiederum dienen als Serviceanbieter für weitere Services der Anwendungsdomäne. Darunter bspw. ein User-Service, der nutzerbezogene Dienste zur Verfügung stellt, oder ein Timing-Service, über den sich beliebige Operationen zeitlich planen lassen.

Insbesondere nehmen die Interface Services Eingaben über den Interface Event Bus entgegen. Sie reichern das Event mit weiteren Informationen, wie Nutzerdaten, an und setzen ein Event im Reactor Event Bus ab. Der hier als Reactor bezeichnete Kern bildet das Herzstück der Anwendung und enthält die eigentliche Anwendungslogik. Die Reactoren offerieren keine Services, sie reagieren lediglich auf Events im Reactor Event Bus. Sie implementieren Algorithmen und Logik zum Treffen von Entscheidungen. Etwa der Navigation Reactor hört auf Nutzerannäherungen und entscheidet anhand verschiedener Faktoren, ob und welche Farbe über die Lichtschnittstelle gezeigt werden soll. Reactoren können alle Services in Anspruch nehmen und auch selbst Reactor Events absetzen. Durch das hier implementierte EDA-Prinzip ist es möglich, die Anwendungslogik flexibel zu erweitern. Mehrere Reactoren können dasselbe Event verarbeiten. Eine Nutzerannäherung kann nicht nur den Navigation Reactor ansprechen, der die Routenführung verarbeitet. Auch ein Reactor für die Komfortzone ist denkbar, der den User Service beansprucht und die Komfortzone des Nutzers anpasst. Der User Service konsultiert hier wiederum die Cloud (nicht eingezeichnet).

Eine besondere Funktion stellt der Broadcasting Service dar. Er hört auf den Reactor Event Bus und erzeugt aus dessen Events MQTT-Nachrichten, die auf das Strahler-eigene Topic veröffentlicht werden. Benachbarte Strahler werden entsprechend über Events benachrichtigt. Auch in anderer Richtung empfängt der Broadcasting Service Events von benachbarten Strahlern und setzt diese in den eigenen Reactor Event Bus ab, damit Reactoren sie verarbeiten können. Des Weiteren implementiert der Broadcasting Service den Propagationsmechanismus, der in 4.3.6 beschrieben wurde.

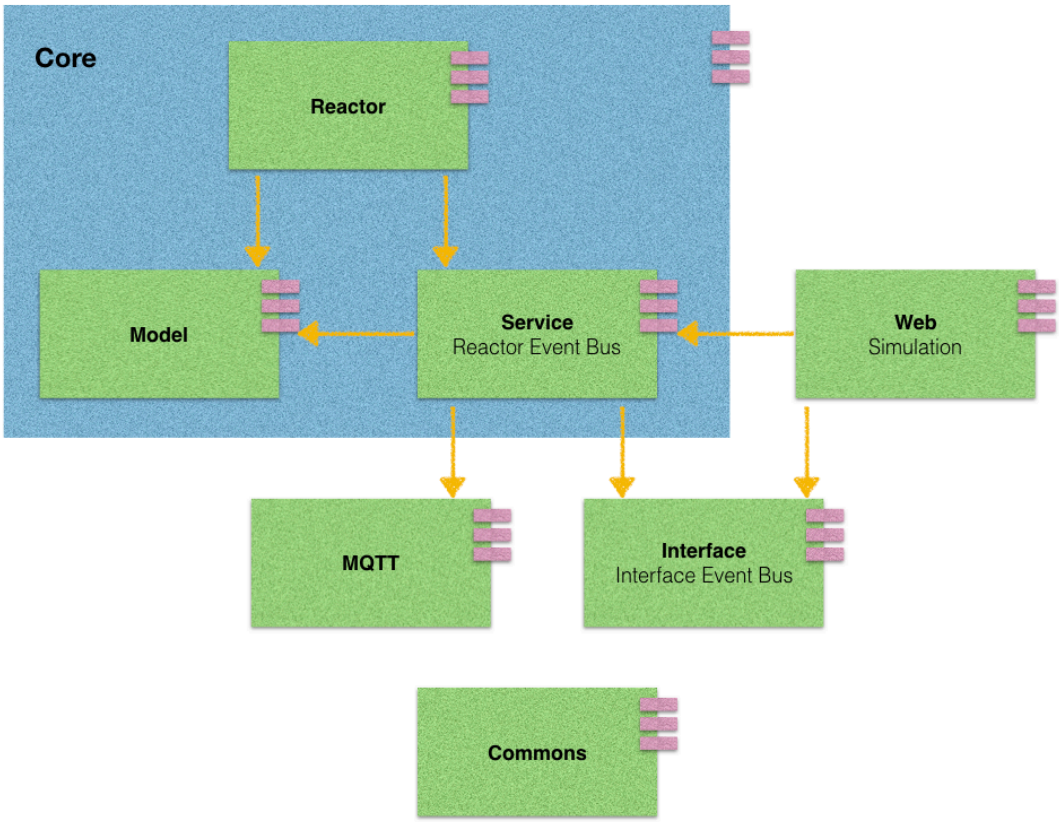

Abbildung 14 - Softwaremodule

Abbildung 14 zeigt die Softwaremodule der multimodularen Codebasis im Stile eines Komponentendiagramms. Einzelne Schnittstellen sind nicht eingezeichnet, die Grafik gibt einen Überblick über die Abhängigkeiten. Die Anwendung ist semantisch und logisch aufgeteilt und die Abhängigkeiten sind wohldefiniert. So kann eine azyklische Softwarearchitektur gewährleistet werden. Die fachliche Logik ist im Core gekapselt. Commons ist eine Bibliothek ohne weitere Projektabhängigkeiten, die von allen Modulen benutzt werden kann[24]. Web ist eine Komponente, die für Simulationszwecke verwendet werden kann. Mehr dazu unter 5.9.

Grundsätzlich finden sich im Komponentendiagramm einige Bausteine wieder, die auch in Abbildung 13 zu sehen sind. Das Modul Model enthält reine Datenklassen (DTOs/POJOs), mit deren Hilfe Reactor und Service sich einheitlich verständigen. Darin liegen auch die Eventtypen, die der Reactor unterstützt.

5.3. Werkzeuge

Der Prototyp wurde mit Hilfe zahlreicher Entwicklungswerkzeuge gebaut, angefangen bei Build-Tools über Deployment-Mechanismen, Betriebssysteme und Middleware.

[24] Auf Grund der Menge wurden die Pfeile, also Abhängigkeiten, zu diesem Modul in der Abbildung nicht dargestellt.

Die Software der MIS ist in Java 8 programmiert und wird mit der Oracle JDK[25] ausgeführt. Als Build-Tool kommt Gradle[26] 3.5 zum Einsatz. Die Modulstruktur der Codebasis (siehe Abbildung 14) ist mit Hilfe von Gradle Subprojects realisiert. Durch das Gradle Dependency Management werden externe Abhängigkeiten beim Build automatisch heruntergeladen. Sie sind nicht Teil der Codebasis. Der Build-Prozess sowie die Abhängigkeiten werden in den `build.gradle` Dateien der (Unter-) Projekte beschrieben. Zum Bauen der Software unter unixoiden Systemen (Linux/Max) genügt es, mit einer Konsole ins Code-Verzeichnis zu navigieren und den folgenden Befehl auszuführen:

```
$ ./gradlew clean build
```

Bis auf Java braucht keine weitere Software installiert zu sein. Der hier genutzte Gradle Wrapper lädt die benötigten Build-Werkzeuge automatisch. Unter Windows-Systemen wird äquivalent die Datei `gradlew.bat` ausgeführt.

Zur Versionskontrolle kommt Git[27] zum Einsatz. Neue Features und Änderungen wurden in Commits festgehalten und damit eine fast lückenlose Versionshistorie gewährleistet. Durch Git-Tags werden stabile Releases markiert. Das Git-Repository ist einerseits in einem Remote Repository der Forschungsgruppe Kooperationssysteme[28] (Origin) gehostet, andererseits auch als Open Source Projekt zugänglich über den GitHub Account des Autors[29]. Die Open Source Lizenz des Softwareprojekts ist die Apache License[30] 2.0, die jedermann reichlich Rechte zur Weiterverwendung einräumt und in der Codebasis abgelegt ist. Ein weiteres Remote-Repository liegt bei Resin.io, siehe weiter unten in diesem Abschnitt.

Die Software wurde in IntelliJ IDEA Ultimate 2017.1 von Jetbrains[31] entwickelt, dies unterstützte den Entwicklungsprozess sehr. Die Software selbst ist jedoch gänzlich unabhängig von einer integrierten Softwareumgebung (engl. kurz IDE) entworfen und kann auch ohne IDE entwickelt werden.

Die in Java entwickelte Software setzt auf das Spring Boot Framework[32] in Version 1.5.4 auf. Dem zugrunde liegt das Spring Framework in Version 4.3.9. Spring Boot bietet diverse wiederverwendbare Komponenten, deren Nutzung die Entwicklung deutlich vereinfacht und beschleunigt. Unter anderem ermöglicht der Spring Boot Kern, dass alle entwickelten MIS-Komponenten (z.B. Service-, Configuration- und Component-Klassen) automatisch beim Anwendungsstart durch Service Discovery gefunden und erzeugt werden. Die Service-

---

[25] http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html (letzter Abruf: 09.08.2017)

[26] https://gradle.org/ (letzter Abruf: 09.08.2017)

[27] https://git-scm.com/ (letzter Abruf: 09.08.2017)

[28] https://bitbucket.cscwlab.de/projects/ULP/repos/mikroinfostrahler/browse (letzter Abruf: 09.08.2017)

[29] https://github.com/vituslehner/ulp_microinforadiator (letzter Abruf: 09.08.2017)

[30] https://www.apache.org/licenses/LICENSE-2.0 (letzter Abruf: 09.08.2017)

[31] https://www.jetbrains.com/idea/ (letzter Abruf: 09.08.2017)

[32] https://projects.spring.io/spring-boot/ (letzter Abruf: 09.08.2017)

Abhängigkeiten untereinander werden per Dependency Injection automatisch aufgelöst und bedient. Das Spring Boot Framework startet im Hintergrund selbstständig einen Apache Tomcat Webserver als Laufzeitrahmen. In den Gradle-Build ist ein Spring Boot Plugin integriert, das den Bau und Start der Software durch einen kurzen Befehl ermöglicht:

```
$ ./gradlew clean build bootRun
```

Die Spring Boot Integration Erweiterung umfasst auch ein MQTT-Modul, das im Prototyp als Client-Bibliothek verwendet wird. MQTT wird zur technischen Abkapselung in einer eigenen Softwarekomponente angebunden, die wiederum vom Broadcasting Service des Cores genutzt wird.

Die Spring Boot Web Erweiterung kommt in der Web-Komponente zum Einsatz. Sie ermöglicht einfache Webanwendungen per HTTP auszuliefern und unterstützt die Entwicklung von REST-Endpunkten stark. Die Web-Komponente ist insbesondere für Testzwecke gedacht, um den Status einzelner Strahler zu beobachten. Sie implementiert eine rudimentäre Weboberfläche. Außerdem angebunden sind hier die Untermodule Thymeleaf als Template Engine, sowie Actuator als REST-Schnittstelle für das Abfragen des Gesundheitszustands und Status der Anwendung.

Die Java-Anwendung läuft potenziell sowohl unter ‚unixoiden'[33], als auch Windows-basierten Betriebssystemen. Auf Grund der freien Verfügbarkeit, Kostenfaktoren und der weiten Verbreitung im IoT-Umfeld ist Linux für diesen Prototyp die Zielplattform. Für Raspberry Pis besonders verbreitet und gut unterstützt ist die Linux-Distribution Raspbian[34], das auf Debian[35] fußt. Der Prototyp nutzt Raspbian der Version Jessie.

Da der Versuchsaufbau und auch der reale Einsatz aus einer Vielzahl von Geräten besteht und die Einrichtung jedes einzelnen Betriebssystems sehr aufwendig ist, werden die Möglichkeiten von Virtualisierung genutzt, um die Geräte aufzusetzen. Docker[36] ist ein Werkzeug, das eher im Umfeld von Cloud Computing bekannt ist, jedoch ermöglicht, virtuelle Maschinen reproduzierbar nach einer Vorlage zu erzeugen. Sogenannte Docker-Images werden mit Hilfe von Dockerfiles beschrieben. Das Dockerfile enthält alle Anweisungen zur Erzeugung eines Images, angefangen bei der Betriebssystembasis, über das Herunterladen und Installieren von Oracle JDK, Gradle und anderer Software bis zur Definition des Start-Befehls. Es ist im Hauptverzeichnis der Codebasis abgelegt. Das Image ist die Vorlage, die als Docker-Container mit individueller Konfiguration (z.B. individuelle Linux-Umgebungsvariablen) gestartet wird. Durch Docker und Docker Compose ist es möglich ganze Scharen an MIS mit wenigen Kommandos zu simulieren.

Durch Docker werden durchgehend einheitliche Umgebungssysteme für die Anwendung erzeugt, allerdings läuft das Verteilen und Starten der Images weiterhin manuell. Resin.io[37] ist ein Dienstanbieter für Deployments im IoT-Bereich, der dieses Problem löst. Der Cloud-Service baut automatisiert Docker-Images auf Basis eines Git-Repositories und verteilt sie an alle Geräte, die

---

[33] Gemeint sind insbesondere die Betriebssysteme Linux und macOS.

[34] https://www.raspbian.org/ (letzter Abruf: 09.08.2017)

[35] https://www.debian.org/ (letzter Abruf: 09.08.2017)

[36] https://www.docker.com/ (letzter Abruf: 09.08.2017)

[37] https://resin.io/ (letzter Abruf: 09.08.2017)

einer Anwendung angehören. Dazu wird in der Webanwendung von Resin.io ein Projekt bzw. eine Anwendung angelegt. Für diese Anwendung wird dann ein Basisbetriebssystem für eine Zielarchitektur erzeugt, hier der Raspberry Pi 3. Dieses Betriebssystemabbild wird auf die microSD-Karte der Raspberry Pis aufgespielt. Es enthält die von Resin.io entwickelte Software zur Verwaltung des Anwendungslebenszyklus. Dieser Schritt ist lediglich initial erforderlich. Beim Start des Raspberry Pi verbindet sich dieser mit der Resin.io Cloud. Nun kann das Gerät aus der Ferne gesteuert und Software ausgeliefert werden. Dazu stellt Resin.io ein eigenes Remote-Git-Repository zur Verfügung, das in die lokale Git-Remote-Liste eingetragen wird. Durch den Befehl

```
$ git push resin master
```

wird der aktuelle Stand der Codebasis an das Resin.io Repository gesendet. Der Resin.io Service liest das Dockerfile und nutzt es als Anleitung zum Bau eines Docker-Images. Im Erfolgsfall wird das Image an alle zur Anwendung gehörenden Geräte übermittelt, also jene Geräte, die mit dem Projekt-spezifischen Betriebssystem bestückt wurden. Dort wird das Image als virtueller Container gestartet. In diesem Fall virtualisiert der Container das Raspbian-Betriebssystem. Über die Webanwendung des Anbieters können die Geräte einzeln konfiguriert werden, indem Betriebssystem-Umgebungsvariablen definiert werden. So ist es bspw. möglich, jedem IS eine individuelle ID-Kennung, seine GPS-Position, seine IS-Nachbarn und den zu verwendenden MQTT-Broker mitzuteilen.

Da die Resin.io Cloud scheinbar nicht auf der gleichen Prozessorarchitektur läuft, wie die Raspberry Pis, wird der Gradle-Build noch nicht beim Erzeugen des Docker-Images ausgeführt, sondern erst beim Container-Start. Dadurch werden etwaige Hardware-abhängige Probleme beim Anwendungsstart verhindert.

Ebenfalls per Git angebunden ist der Continous Integration Dienst von Travis CI[38], der für Open Source Projekte kostenlos und durch GitHub leicht zu integrieren ist. Im Hauptverzeichnis ist dazu die Datei `.travis.yml` angelegt, die eine kurze Instruktion zum Build des Projekts enthält. Bei jedem Push auf das GitHub Repository wird automatisch ein Build initiiert, in dem auch etwaige Unit- oder Integration-Tests ausgeführt werden. Dies ermöglicht eine kontinuierliche Fehler- und Qualitätskontrolle. Außerdem erkennt Travis CI Git-Tags und löst für sie automatisch Release-Builds aus. Die dabei entstehenden (ausführbaren) Artifakte werden selbstständig auf der GitHub Release Seite[39] eingetragen und verknüpft.

Auf tieferer, Schnittstellen-naher Ebene wurden einige weitere Technologien eingesetzt. Zur Realisierung der iBeacons wurde die BlueZ[40]-Bibliothek verwendet, die unter Linux-Systemen als verbreitet gilt. Die Bibliothek wird in Java im Interface-Modul durch einen dafür erzeugten Konsolen-Prozess genutzt. Darüber werden auch iBeacon-Details wie UUID, Major und Minor übergeben. Die von und für BlueZ benötigte Software wird im Dockerfile definiert und durch den Raspbian-Paketmanager `apt-get` installiert.

---

[38] https://travis-ci.org/vituslehner/ulp_microinforadiator (letzter Abruf: 09.08.2017)

[39] https://github.com/vituslehner/ulp_microinforadiator/releases (letzter Abruf: 09.08.2017)

[40] http://www.bluez.org/ (letzter Abruf: 09.08.2017)

Auch die Macher des Sense HAT Moduls liefern keine Implementierung der Schnittstelle für Java, jedoch eine Python-Bibliothek[41] zum Steuern des Moduls. Hier wurde eine kleine Bash-Anwendung geschrieben, die aus der Java-Anwendung heraus in einem eigenen Prozess gesteuert wird. Auch die Python- und Sense HAT-Abhängigkeiten werden über das `apt-get` des Dockerfiles installiert.

MaryTTS[42] ist ein Text-to-Speech-System (kurz TTS-System) mit Mehrsprachfähigkeiten. Es steht als Open Source zur Verfügung. Im Prototyp kommt es für die Sprachsynthetisierung und -ausgabe zum Einsatz. Das System kann einfache Text-Strings entgegennehmen und synthetisiert daraus eine Stimme. Die Stimme kann konfiguriert werden, hier wird eine deutsche Vorlage verwendet. Da MaryTTS in Java entwickelt wurde und auch direkt in Java-Projekte integrierbar ist, wird hier kein Umweg über ein zusätzliches Script benötigt, um Sprachausgaben durch das Speech Interface zu erzeugen.

Esri[43] ist ein Softwarehersteller von Geoinformationssystemen, der eine Geometrie-Bibliothek[44] für Java als Open Source bereitstellt. Die Bibliothek wird vom Prototypen für geometrische Berechnungen mit bspw. Koordinaten und Pfaden genutzt.

Wie weiter oben bereits beschrieben, wird als Client-Bibliothek ein Spring Boot Modul für die MQTT-Integration genutzt. Darüber hinaus ist für die Funktion auch ein MQTT-Broker als Middleware nötig. Hier gibt es verschiedene Softwarelösungen verschiedener Anbieter. Mosquitto[45] ist eine Open Source Implementierung des Protokolls unter dem Dach der Eclipse Foundation und ist für prototypische Einsätze geeignet. Es gibt auch fertige Docker-Images, die genutzt werden können. Im Versuchsaufbau wurde zur Vereinfachung der öffentliche MQTT-Broker der Eclipse Foundation[46] unter Host `iot.eclipse.org` und Port `1883` genutzt. Da jeder dessen Nachrichten einsehen kann, sollten keine realen oder vertraulichen Daten darüber übertragen werden.

Der Vollständigkeit halber wird noch erwähnt, dass die Google Guava[47] Bibliothek als allgemeines Mittel bei der Programmierung genutzt wurde. Sie stellt einige häufig gebrauchte Methoden bereit, etwa zur Exception-Behandlung oder String-Manipulation. Aus Guava wird insbesondere auch die Event Bus Implementierung genutzt, die für Interfaces und Reactor eingesetzt wird. Zur (De-) Serialisierung von JSON und XML wird die Jackson-Bibliothek[48] verwendet.

---

[41] https://pythonhosted.org/sense-hat/ (letzter Abruf: 09.08.2017)

[42] http://mary.dfki.de/ (letzter Abruf: 10.08.2017)

[43] http://www.esri.com/ (letzter Abruf: 11.08.2017)

[44] https://github.com/Esri/geometry-api-java (letzter Abruf: 11.08.2017)

[45] https://mosquitto.org/ (letzter Abruf: 09.08.2017)

[46] https://iot.eclipse.org/getting-started#sandboxes (letzter Abruf: 09.08.2017)

[47] https://github.com/google/guava (letzter Abruf: 09.08.2017)

[48] https://github.com/FasterXML/jackson (letzter Abruf: 09.08.2017)

5.4. MQTT-Topic-Architektur

Es gibt einige Praktiken bei der Wahl von MQTT-Topics. Topics sind hierarchisch aufbaubar, ähnlich einer Baum- oder Ordner-Struktur. Das Topic kann dabei beliebig viele Ebenen haben. Ebenen werden durch einen Schrägstrich getrennt. Beispiel: `device/12345/sensors/airquality`.

Für den Prototyp wurde als Wurzel das Kürzel `ulp`, für UrbanLife+, verwendet. Topics beginnen also mit `ulp/`. MIS veröffentlichen Events hauptsächlich auf der zweiten Ebene `mir/` (Micro Information Radiator). Smartphones veröffentlichen Events auf der zweiten Ebene `sp/` (SmartPhone).

IS veröffentlichen ihre eigenen Reactor-Events und propagierte Events auf ihr IS-spezifisches Topic. Dazu wird auf dritter Ebene die IS-ID eingesetzt. Das Topic lautet also etwa `ulp/mir/<is-id>`. `<is-id>` ist dabei ein Platzhalter für eine entsprechende ID.

Das wichtigste Event, das Smartphones veröffentlichen, ist die Annäherung eines Nutzers. Hier enthält das Topic sowohl Nutzer-ID, als auch IS-ID auf verschiedenen Ebenen. Das Topic lautet `ulp/sp/proximity/<user-id>/<is-id>` bzw. `ulp/sp/proximity/<user-id>/<ibeacon-major>-<ibeacon-minor>`. Dass das Topic sowohl die Nutzer-ID, als auch Strahler-ID enthält, ist insbesondere für das Sicherheitskonzept wichtig. Die MQTT-Autorisierung basiert auf Ressourcenzugriffskontrolle, und die Ressourcen sind hier die Topics. So kann gewährleistet werden, dass nur Smartphones eines bestimmten Nutzers auf dessen Topic veröffentlichen können und andererseits nur Strahler mit der gegebenen ID das Topic abonnieren können. Dabei ist nicht nötig, dass Strahler das Topic jeden Users abonnieren. Hier kann mit Wildcards gearbeitet werden. `+` ist ein Platzhalter für eine Ebene. `#` kann beliebig viele Ebenen ersetzen, kann jedoch nur am Ende eines Topics verwendet werden. Die Autorisierungskonfiguration würde in diesem Fall dem Nutzer-Smartphone erlauben auf alle Topics des Typs `ulp/sp/proximity/<user-id>/+` zu veröffentlichen, IS würden alle Topics des Schemas `ulp/sp/proximity/+/<is-id>` abonnieren dürfen.

5.5. Funktionsweise

Der Prototyp soll die Funktionalität der Routenführung beherrschen, dabei bildet die im Konzept vorgestellte Komponentenarchitektur mit SOA und EDA die Basis. Für die konkrete Implementierung bleibt dabei noch Freiraum, so wird nachfolgend erläutert, wie der Prototyp die Funktion implementiert und welche Möglichkeiten dabei abgewogen werden.

5.5.1. Routenführung

Das in der Architektur vorgestellte EDA-Reaktionsprinzip kommt in der Routenführung zum Einsatz. Einzelne MIS reagieren dabei auf Ereignisse, die in ihrer Umgebung auftreten. Besonders relevant ist dabei das Event, bei welchem sich ein Nutzer an einen MIS (nachfolgend IS1) annähert.

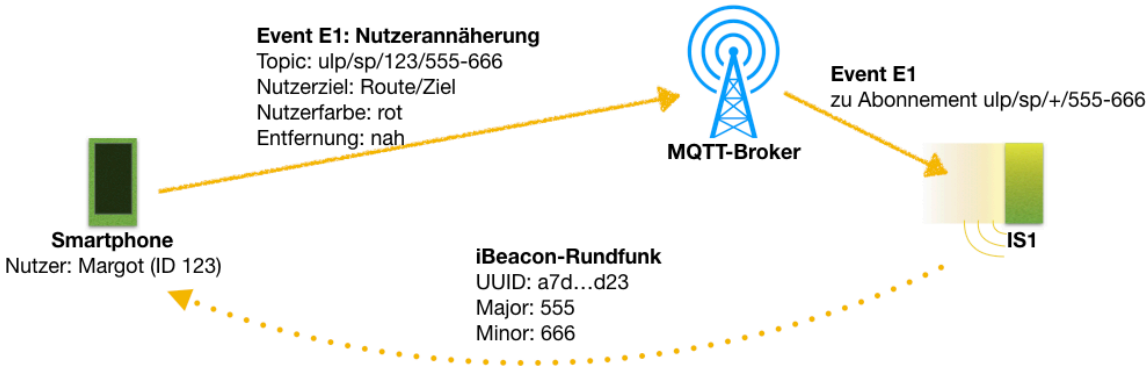

Abbildung 15 - Prototyp: Routenführung Schritt 1

Abbildung 15 zeigt den ersten Schritt einer solchen Annäherung. IS1 als iBeacon-Transmitter sendet laufend seine iBeacon-Details aus. Das Nutzersmartphone empfängt diese und evaluiert sie. Gehört die ausgestrahlte UUID zur UrbanLife+-Anwendung, so reagiert es auf den Strahler. Es verbindet sich mit einem MQTT-Broker. Dort veröffentlicht es ein Nutzerannäherungsevent E1 und wählt das Topic so, dass es den Major und Minor von IS1 enthält. IS1 ist Abonnement dieses Topics, so stellt der Broker E1 dem Strahler zu.

Die Eventnachricht kann diverse Informationen enthalten, die ein Strahler für eine Reaktion verarbeiten kann. Die Nachricht in der Abbildung enthält alle Informationen, die IS1 für die Aufgabe der Routenführung benötigt. Neben der Nutzer-ID (im Topic) ist dies besonders eine nutzerspezifische Farbe für die Lichtfarbe der Strahler, und das Ziel bzw. die Route des Nutzers. Für den Fall, dass der Nutzer in dem Zeitpunkt kein Ziel gesetzt hat, kann dieses Feld ausgelassen oder leer bleiben.

Hier nicht eingezeichnet, technisch aber möglich und – wenn vom Nutzer gewünscht – hilfreich ist auch die Übermittlung von sensitiven Daten, bspw. körperliche Beeinträchtigungen, die IS beachten sollen. An dieser Stelle ist eine Datenschutzweiche denkbar. Nutzern kann ermöglicht werden am Smartphone selbst zu konfigurieren, welche Informationen an Informationsstrahler weitergegeben werden sollen, unter Inkaufnahme der Folge, dass IS Strahler nicht im gleichen Maße unterstützen können.

IS1 nimmt das Event E1 entgegen, nutzt die Nutzer-ID um durch den User Service weitere Informationen aus der Cloud zu laden und reichert das Event mit Informationen an. Das Event wird dann in den internen Reactor Event Bus abgesetzt und steht damit Reactoren zur Verfügung. Reactoren wie der Navigation Reactor, die diesen Typ von Nachricht abonniert haben, verarbeiten das Event. Der Navigation Reactor enthält in diesem Fall Anwendungslogik für das Annäherungsevent und entscheidet darüber, wie sich IS1 nachfolgend verhält. Dazu nutzt er die Informationen, die ihm durch das Event zur Verfügung gestellt werden.

Zentraler Entscheidungspunkt für IS1 ist, ob der Nutzer sich auf seiner Route befindet oder nicht, damit entsprechend ein bestätigendes oder ablehnendes Ton- und Lichtsignal gegeben werden kann. IS1 steht dafür die aktuelle Nutzerposition[49], sowie Informationen zur Route bzw. dem Ziel des Nutzers zur Verfügung. An dieser Stelle und auch in der Abbildung blieb bislang ungeklärt, wie dieser Informationsparameter (Nutzerziel) des Events E1 konkret gestaltet ist, sprich welche

[49] Es wird davon ausgegangen, dass die Nutzerposition in diesem Moment der Position des Strahlers entspricht. Diese Position ist im Strahler in Form von Koordinaten konfiguriert.

Routendetails das Smartphone an den Strahler übermittelt. Denkbar sind das Ziel des Nutzers selbst (kodiert als Koordinaten, ID, Adresse o.ä.) oder die Route selbst (kodiert als Abfolge von Wegpunkten bzw. Wegkoordinaten).

Der Prototyp nutzt geometrische Funktionen, um zu ermitteln, ob der Standort eines Strahlers auf einer Route liegt. Hierbei ist der Standort gegeben als Paar aus Längen- und Breitengrad, die Route gegeben als Liste solcher Paare. Erhält der Informationsstrahler einen Zielort als Nutzerziel, so kann er von einem externen Kartenservice eine Route abfragen und die Wegpunkte extrahieren. Erhält der Strahler die Wegpunkte direkt vom Smartphone, so entfällt dieser Schritt. Dabei entsteht ein weiterer Vorteil zum Schutz der Privatsphäre: Das Smartphone muss nicht alle Wegpunkte bis zum Nutzerziel übermitteln. Es besteht die Möglichkeit, lediglich den Teil einer Route zu übertragen, der innerhalb eines gewissen Radius liegt und bis zu den nächsten zwei bis drei Informationsstrahlern reicht. Der letzte Wegpunkt einer übertragenen Route ist ggf. also nur ein Zwischenziel. Dadurch erfahren nicht alle IS vom tatsächlichen Ziel des Nutzers.

Im Prototyp ist der Reactor auf Basis der Annahme implementiert, dass das Smartphone das Nutzerziel als Route, im Sinne einer Liste von Wegpunkten übermittelt. Mit der räumlichen Geometrie-Bibliothek Esri für Java wird bestimmt, ob der Standort des Strahlers auf der Route liegt. Ist dem so, pulsiert der Strahler einmalig in grüner Farbe und spielt einen Bestätigungston ab. Ist der Nutzer von seiner Route abgekommen, macht er dies durch ein entsprechendes Symbol bemerkbar und spielt einen Fehlerton, gefolgt von der Aussage, dass der Nutzer vom Weg abgekommen ist.

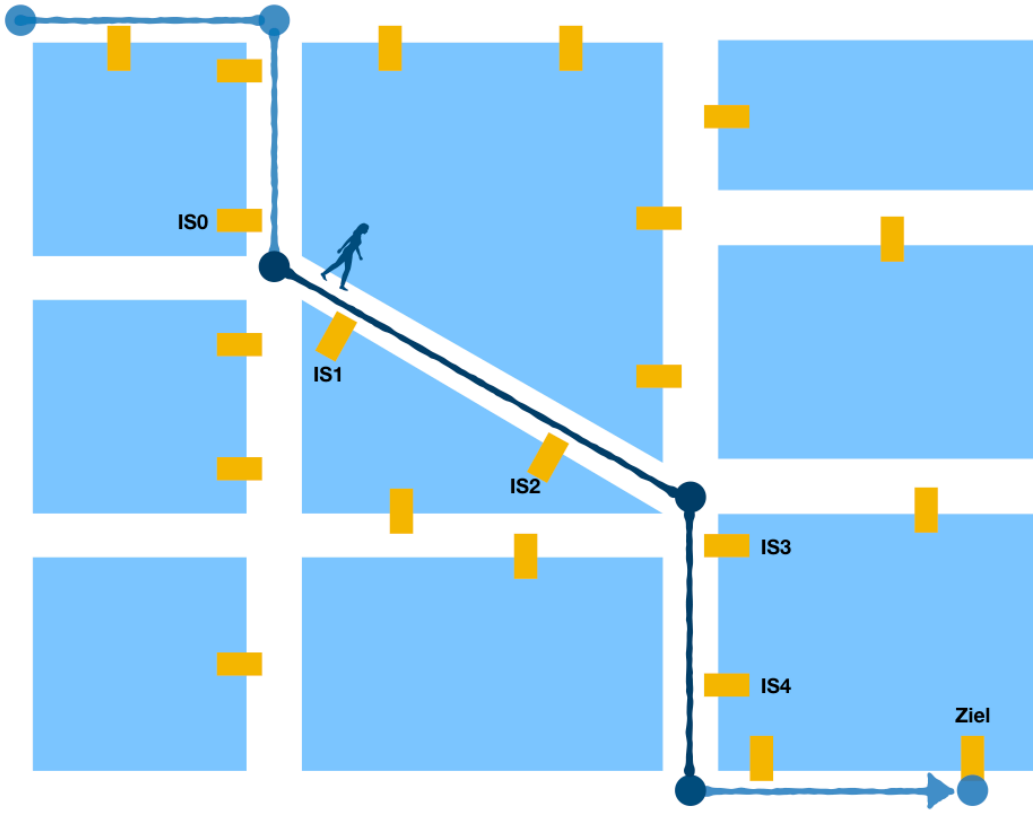

Abbildung 16 - Szenario Routenführung aus Vogelperspektive

Abbildung 16 zeigt anschaulich das vorliegende Szenario einer Routenführung aus räumlicher Sichtweise: der Vogelperspektive. Eingezeichnet sind einige Straßen (weiß) zwischen Blöcken,

wie Gebäuden oder Grünanlagen (blau). Mikroinformationsstrahler sind in der Abbildung als gelbe Rechtecke an Gehwegen eingezeichnet, teils benannt. Dunkelblau ist die vom Nutzer angeforderte Route eingezeichnet, die durch runde Wegpunkte gegeben ist. Das Ziel befindet sich unten rechts. Im Zeitpunkt dieses Schnappschusses befindet sich der Nutzer an IS1, wo bereits ein entsprechendes Annäherungsevent verarbeitet wird. Das Smartphone hat lediglich die Wegpunkte vom Standort des Nutzers bis zu einem Wegpunkt in gewisser Entfernung dem Annäherungsevent mitgegeben (hier besonders dunkel dargestellt). Diese Wegpunkte sowie seine Koordinaten nutzt IS1, um zu ermitteln, ob sich der Nutzer auf der Route befindet. In diesem Fall trifft dies zu. Da der IS-Standort aus geometrisch-mathematischer Sicht nicht exakt auf der Linie liegt, wird hier eine geeignet gewählte Toleranz gewählt. Man könnte sagen, die Linie wird angemessen verdickt.

Das in den Reactor Event Bus von IS1 abgesetzte Event wird auch vom Broadcasting Service gelesen. Dieser reichert das Event mit einigen zusätzlichen Daten an und veröffentlicht das Event auf das IS1-eigene Topic. IS1 habe hier die ID 1. Die Strahler IS0 und IS2 sind direkt benachbart zu IS1. Sie haben das Topic abonniert und bekommen das Event P1 vom Broker zugestellt.

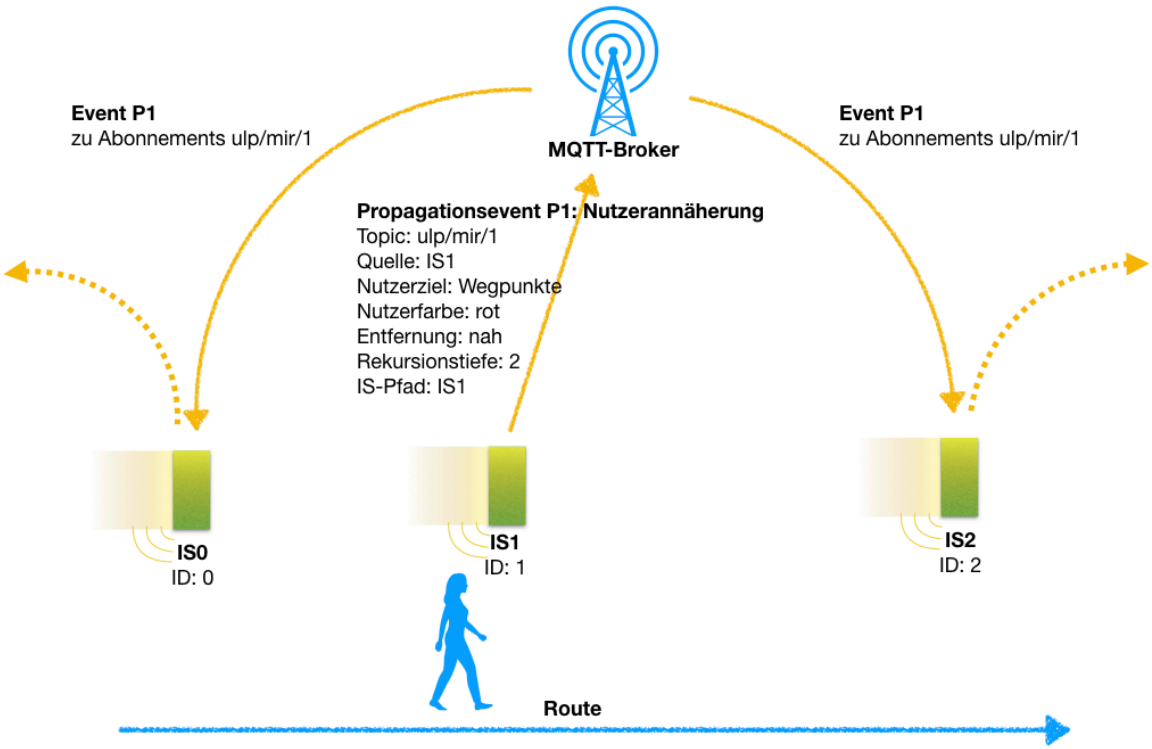

Abbildung 17 - Prototyp: Routenführung Schritt 2

Abbildung 17 zeigt, wie sich die propagierende Eventnachricht P1 verbreitet. Über den in 4.3.6 vorgestellten Rekursionsmechanismus erreicht das Event zunächst IS0 und IS2. Gestrichelt verdeutlicht ist, dass P1 je nach gegebener Rekursionstiefe auch noch andere IS erreichen kann. Auch dies würde über den MQTT-Broker abgewickelt werden, ist zur Vereinfachung hier jedoch mit direkten, gestrichelten Pfeilen dargestellt.

Neben den Informationen, die das Smartphone bereits in Event E1 bereitgestellt hat, ist P1 ergänzt um eine Quelle. Die Quelle ist die ursprüngliche Herkunft eines Propagationsevents im Sinne eines Informationsstrahlers, hier IS1. Die Quelle bleibt bei jedem Propagationsschritt gleich.

Auch für IS0 und IS2 ist das empfangene Event vom Typ ‚Nutzerannäherung'. Das Event wird in die jeweiligen Reactor Event Busse eingespeist und ebenfalls von den einzelnen Navigation Reactoren verarbeitet. Die Quelle kann an dieser Stelle genutzt werden, um zu determinieren, ob der Nutzer diesem Strahler (IS0 bzw. IS2) nahegekommen ist, oder einem anderen. In diesem Fall befindet sich der Nutzer an IS1, damit kommen IS0 und IS2 zum Schluss, dass der Nutzer ihnen nicht nah ist – also fern. Sie geben deshalb kein unmittelbares visuelles oder auditives Feedback.

IS0 und IS2 führen die selbe Logik aus, wie IS1, um zu ermitteln, ob sie auf der Route des Nutzers liegen. IS0 befindet sich nicht (mehr) auf der Route, deshalb sorgt sein Navigation Reactor dafür, dass sein Licht ausgeschalten wird. IS2 hingegen befindet sich auf dem Pfad, deshalb beginnt er rot zu leuchten bzw. deshalb bleibt er rot leuchtend.

Das Beispiel ist mit einer Rekursionstiefe von 2 illustriert, somit erreicht Event P1 nebst anderen auch IS3. Wie IS2 liegt dieser auf dem Pfad, so beginnt auch er rot zu leuchten.

Ebenfalls mit Hilfe von Geometrie ist bestimmbar, in welche Richtung ein Strahler einen Pfeil zu zeigen hat – links oder rechts. Dabei wird der geradlinige Pfadabschnitt ermittelt, an dem der jeweilige IS liegt. Gemeinsam mit konfigurierten Details über Straßenseite der Montage und Himmelsausrichtung des Strahlers, lässt sich nun entscheiden, ob ein Pfeil in Richtung links oder rechts dargestellt werden muss. Der entwickelte Prototyp kann zu illustrativen Zwecken Pfeile in beiden Richtungen darstellen, umfasst jedoch keine Logik, um ihre Richtung zu bestimmen.

Der Prototyp unterstützt außerdem die Routenführung mehrerer Nutzer. Dies ist insbesondere relevant, wenn sich zwei Routen kreuzen oder parallel verlaufen. Die beiden übermittelten Nutzerfarben werden dann abwechselnd gezeigt. Im Fall der farblichen Ähnlichkeit bleibt den Nutzern offen, am Smartphone eine andere Farbe zu konfigurieren.

Die hier vorgestellten Paradigmen basieren darauf, dass eine Route durch Wegpunkte gegeben ist, wie sie bspw. von der Google Maps API[50] kommen. Während einige Tests des Autors gezeigt haben, dass dies durchaus funktioniert, zeigen sich unter Randbedingungen jedoch auch Probleme. Dies ist zum Beispiel der Fall, wenn eine Straße sehr breit ist, die Wegpunkte zentriert verlaufen und seitlich liegende IS nicht in den Toleranzbereich der Route fallen. Eine bessere Funktionsweise wäre hier gegeben, wenn die vom Smartphone übermittelten Wegpunkte genau den Koordinaten der Informationsstrahler entsprächen. Für die Routenberechnung am Smartphone, an einer Informationstafel oder in der Cloud müsste dann jedoch ein gesonderter Zielführungsalgorithmus angewandt werden, der nicht Teil dieser Arbeit ist.

5.5.2. Sprachhinweise

Der Prototyp soll Nutzern unabhängig oder abhängig von einer Routenführung auch situationsbezogene sprachliche Hinweise unterbreiten. Dabei kommen teils (simulierte) externe Quellen und eine gewisse Randomisierung zum Einsatz.

An dieser Stelle kann die Integration des Safety-Atlas einen wesentlichen Mehrwert für Nutzer bieten, indem dessen Daten genutzt werden um auf Hürden oder Beeinträchtigungen der Barrierefreiheit hinzuweisen. Da der Safety-Atlas noch nicht in integrierbarer Form existiert, wird

[50] https://developers.google.com/maps/?hl=de (letzter Abruf: 11.08.2017)

ein externer Wetterdienst als Anbindung simuliert. Er wird simuliert, um das Wetter in Tests selbst einstellen zu können.

Ein Weather Reactor nutzt die Wettervorhersage, um algorithmisch Empfehlungen zu generieren. Droht es zu regnen oder zu gewittern, so warnen Informationsstrahler durch eine Sprachausgabe, wenn Nutzer diesem näher kommen. Damit die Aussage nicht von jedem Strahler erzeugt wird, wird ein Zufallsparameter eingebaut. Des Weiteren wird die Sprachausgabe in mehreren Versionen abgewandelt, um nicht einfältig zu wirken.

Dem Nutzer könnten an dieser Stelle auch informativere und individuellere Empfehlungen gemacht werden, doch wie schon in vorherigen Abschnitten wiederholt festgestellt, steht dies oft im Konflikt mit dem Schutz personenbezogener Daten. Der Prototyp unterstützt auch die Verarbeitung körperlicher Beeinträchtigungen, so werden Sprachausgaben bspw. lauter wiedergegeben, wenn der Nutzer schwerhörig ist. Es liegt am Nutzer bzw. dessen Smartphone solche Informationen mit an den Informationsstrahler zu übergeben. Die Implementierung des Prototyps ist hier ausreichend flexibel, um mit fehlenden Informationen kulant umzugehen.

5.6. Datenmodell

Das in dieser Software zentrale Datenobjekt stellt das Annäherungsevent dar, das für die Reactor Event Busse der einzelnen Informationsstrahler mit Informationen angereichert und in diesen abgesetzt wird. Abbildung 18 zeigt als UML-Klassendiagramm, wie sich das Event zusammensetzt: aus der Strahler-Quell-ID, Benutzerdetails und der eigentlichen Nähe. Die abstrakte Klasse `ReactorEvent` stellt hier einen Rahmen für eine Vielzahl unterschiedlicher, konkreter Events zu Verfügung. Hier ist es das `UserProximityEvent`. Während der Umfang des Events einen überschaubaren Eindruck macht, ist zu bedenken, dass die Klasse User noch einige für die Anwendung nützliche Details enthält.

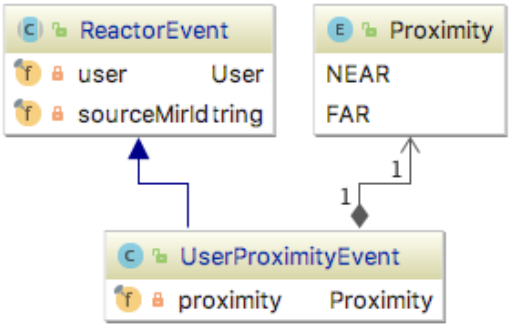

Abbildung 18 - Klassendiagramm: Annäherungsevent

Die Klasse `User` enthält im Prototypen nur so viele Attribute, wie für die prototypischen Zwecke benötigt. Da einige der in ihr enthaltenen Informationen vom CommunityMashup geladen werden (Item-Ident, Vorname, Nachname, etc.), gilt die Klasse `Person`[51] als Vorlage. Tatsächlich könnte diese Klasse also auch Beziehungen zu Organisationen, anderen Personen oder (Aktivitäts-)

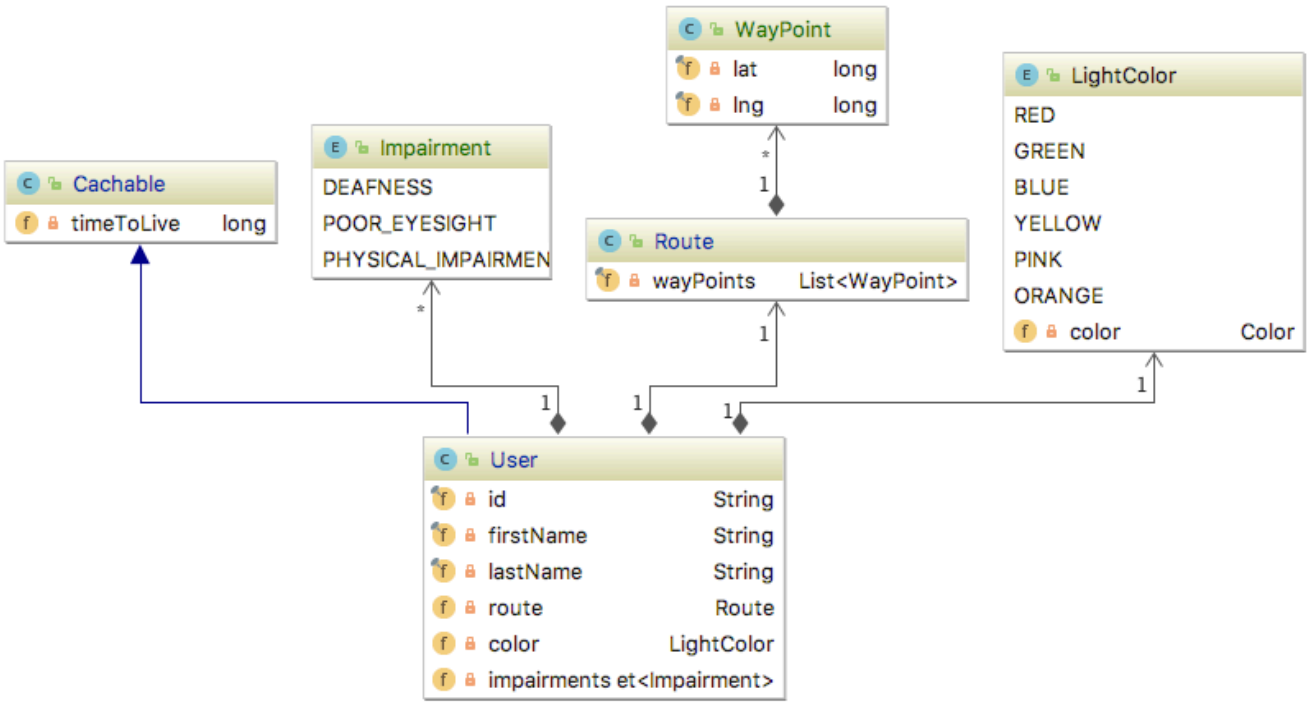

Abbildung 19 - Klassendiagramm: User

Teilnahmen enthalten. `User`-Objekte werden darüber hinaus angereichert mit Informationen, die vom Smartphone kommen: Route, Lichtfarbe und Beeinträchtigungen. Mit `E` gekennzeichnet sind Klassen, die Java-Enums sind – hier `Impairment` und `LightColor`. Beide sind gegeben durch ein festes Set an möglichen Ausprägungen. Die `Route` ist primär durch eine Liste von Wegpunkten (`WayPoint`) gegeben, die jeweils aus Längen- und Breitengraden bestehen.

Das Klassendiagramm zeigt darüber hinaus die abstrakte Klasse `Cachable`, von welcher `User` erbt. Das Attribut `timeToLive` wird hier genutzt, um `User`-Objekte für eine gewisse Zeit zu cachen. Der `UserService` hebt die Objekte auf, so sind sie ggf. für eine spätere Verarbeitung nochmal zugänglich. Des Weiteren müssen sie nicht für jedes Event aus der Cloud geladen werden.

---

51 https://github.com/soziotech/CommunityMashup/blob/master/core/CommunityMashupCore/src/org/sociotech/communitymashup/data/Person.java (letzter Abruf: 12.08.2017)

Abbildung 20 zeigt die im Prototyp genutzt Klasse einer MQTT-Nachricht: `MqttMessage`. Von und zu dieser Klasse werden die als JSON kodierten Nachrichten (de-) serialisiert. Das Attribut `rawData` enthält hier das eigentliche Eventobjekt. Auf dieser Ebene der Übertragung sind auch die in 4.3.6 vorgestellten Parameter für Rekursionstiefe und IS-Pfad enthalten. Darüber hinaus auch die ID des Quell-Strahlers, sowie das MQTT-Topic. Für die Deserialisierung ist außerdem das Attribut `className` wichtig. Es enthält Package- und Klassenname in Form eines `String`. Die empfangende Komponente kann anhand dessen aus dem JSON-String in `rawData` wieder ein Objekt erzeugen. Im Beispiel des Annäherungsevents lautet der `className`: `org.sociotech.urbanlifeplus.microinforadiator.model.event.UserProximityEvent`.

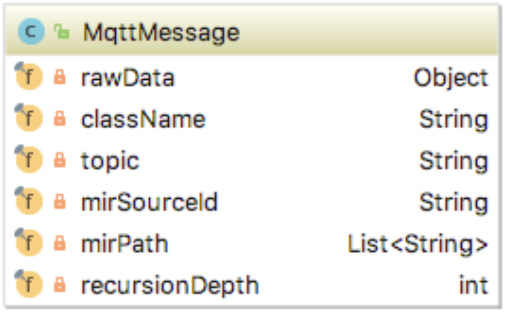

Abbildung 20 - Klassendiagramm: MQTT-Nachricht

5.7. Versuchsaufbau

Beim Testen des Prototyps wurden je nach Bedarf unterschiedliche Setups aufgefahren. Als echte Geräte stehen drei Raspberry Pi 3 mit je montiertem Sense HAT Modul zur Verfügung. Alle drei werden automatisch durch Resin.io deployed, sobald Änderungen am Code auf das Git Remote Repository gepusht werden. Die Geräte bieten sich besonders für Tests der Licht- und Tonausgabe an. Abbildung 21 zeigt die drei leuchtenden Geräte. Sie stellen auch Pfeile dar, was auf dem Foto leider schlecht zu erkennen ist.

Abbildung 21 - Raspberry Pis des Versuchsaufbaus

Für Einzeltests der Software wird die Java-Anwendung auch auf dem lokalen Laptop hochgefahren. Die Interfaces funktionieren dort nur zum Teil, jedoch lässt sich der Core hier besser testen und debuggen.

Für größere Tests kann Docker genutzt werden, um die Software lokal oder entfernt mehrfach zu starten. Dabei werden Mikrostrahler sozusagen simuliert. Für jeden Strahler wird eine eigene virtualisierte Umgebung gestartet. Um eine große Menge an Strahlern zu simulieren, könnten sie auch in der Cloud gestartet werden. Durch Docker Compose lässt sich außerdem eine Menge von Docker-Containern per Konfiguration definieren, so kann per einzelnem Konsolenbefehl ein ganzer Schwarm von Strahlern gestartet werden, die jeweils individuell konfiguriert sind.

5.8. Konfiguration

Die Informationsstrahler müssen zur ordnungsgemäßen Funktion konfiguriert werden, dies geschieht durch Umgebungsvariablen des Betriebssystems (engl. Environment Variables, kurz EV). Die EVs der an Resin.io angebundenen Raspberry Pis lassen sich über die Resin.io-Webanwendung definieren (siehe Abbildung 22).

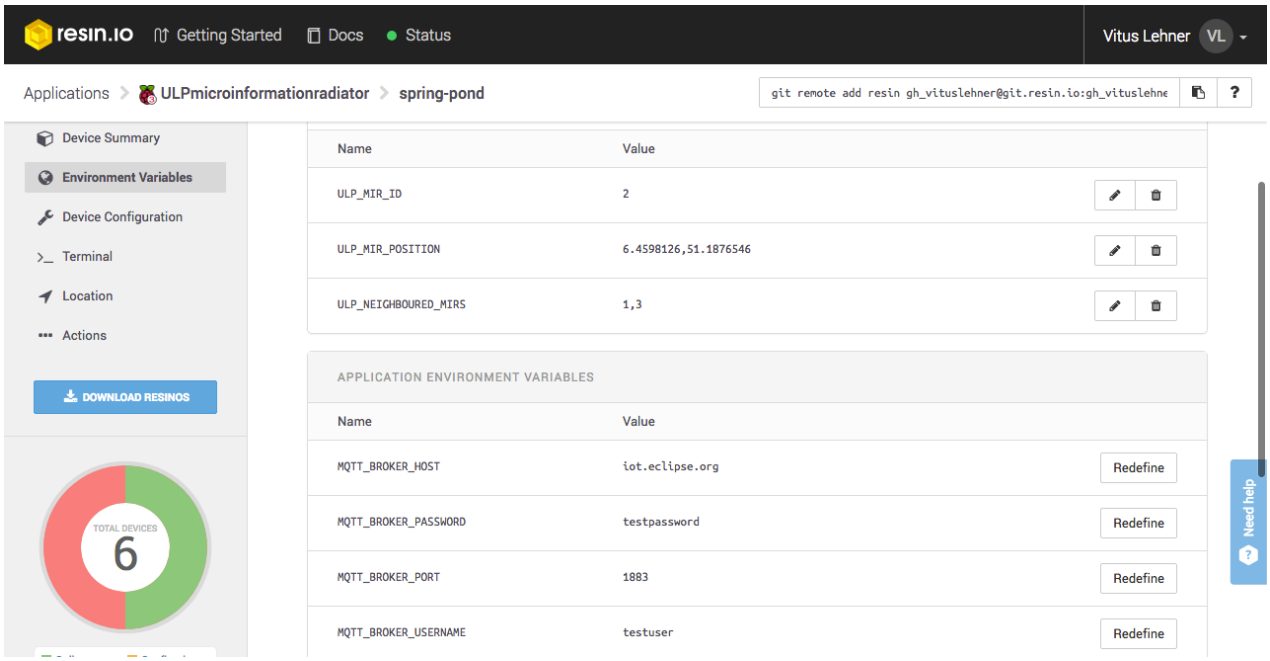

Abbildung 22 - Bildschirmfoto: EV-Konfiguration eines MIS in Resin.io-Webanwendung

Auch mit Docker und Docker Compose lassen sich beim Start von Containern EVs setzen. Startet man die Java-Anwendung außerhalb eines Docker-Containers, so lassen sich die Variablen auch als Java System Properties mitgeben.

Im Genaueren unterstützt der Prototyp die folgenden Konfigurationsparameter:

- `ULP_MIR_ID`:
  ID des MIS
- `ULP_MIR_POSITION`:
  Installationsort des MIS, gegeben durch Längen- und Breitengrad mit Komma getrennt
- `ULP_NEIGHBOURED_MIRS`:
  durch Komma getrennte ID-Liste unmittelbar benachbarter MIS
- `ULP_RECURSION_DEPTH`:
  numerische, initiale Rekursionstiefe für Eventpropagation
- `ULP_PUSH_STATUS`:
  boolescher Wert, ob IS-Status regelmäßig übermittelt werden soll (für Testzwecke)
- `MQTT_BROKER_HOST`:
  Hostname oder IP des zu verwendenden MQTT-Brokers
- `MQTT_BROKER_PORT`:
  Port des MQTT-Brokers
- `MQTT_BROKER_USERNAME`:
  gegenüber dem Broker zur Authentisierung zu nutzender Username
- `MQTT_BROKER_PASSWORD`:
  gegenüber dem Broker zur Authentisierung zu nutzendes Passwort
- `ULP_MIR_IBEACON_UUID`:
  die auszustrahlende iBeacon-UUID (i.d.R. für alle Strahler gleich)
- `ULP_MIR_IBEACON_MAJOR`:
  der auszustrahlende iBeacon-Major

- `ULP_MIR_IBEACON_MINOR`:
  der auszustrahlende iBeacon-Minor
- `CM_BASE_URL`:
  die Basis-URL der zu verwendenden CommunityMashup-Instanz

Die Konfiguration des Internetzugangs erfolgt im Falle der Resin.io-gesteuerten Raspberry Pis nicht durch EVs, sondern muss an der Hardware selbst vorgenommen werden. Dazu wird die WLAN-Konfiguration auf den microSD-Karten der Geräte hinterlegt. Alternativ können die Strahler durch ein LAN-Kabel angebunden werden.

5.9. Simulation

Um die Funktion des Prototyps bzw. der Prototypen darüber hinaus zu testen und ihre Funktionsweise vorzuführen, wurde außerdem das Softwaremodul `Web` entwickelt. Es stellt eine einfache, kleine Website via HTTP zur Verfügung und erlaubt den Status verschiedener Strahler einzusehen, sowie Ereignisse zu simulieren. Die Website nutzt die im Eclipse Paho Projekt entwickelte JavaScript-Bibliothek[52] einer MQTT-Client-Implementierung. Darüber hinaus wurde das IS-übergreifendes Topic `ulp/mir/status` eingerichtet, auf welches alle IS regelmäßig einen Status veröffentlichen. Durch die EV `ULP_PUSH_STATUS` lässt sich diese Funktion auch ausschalten.

[52] https://eclipse.org/paho/clients/js/ (letzter Abruf: 12.08.2017)

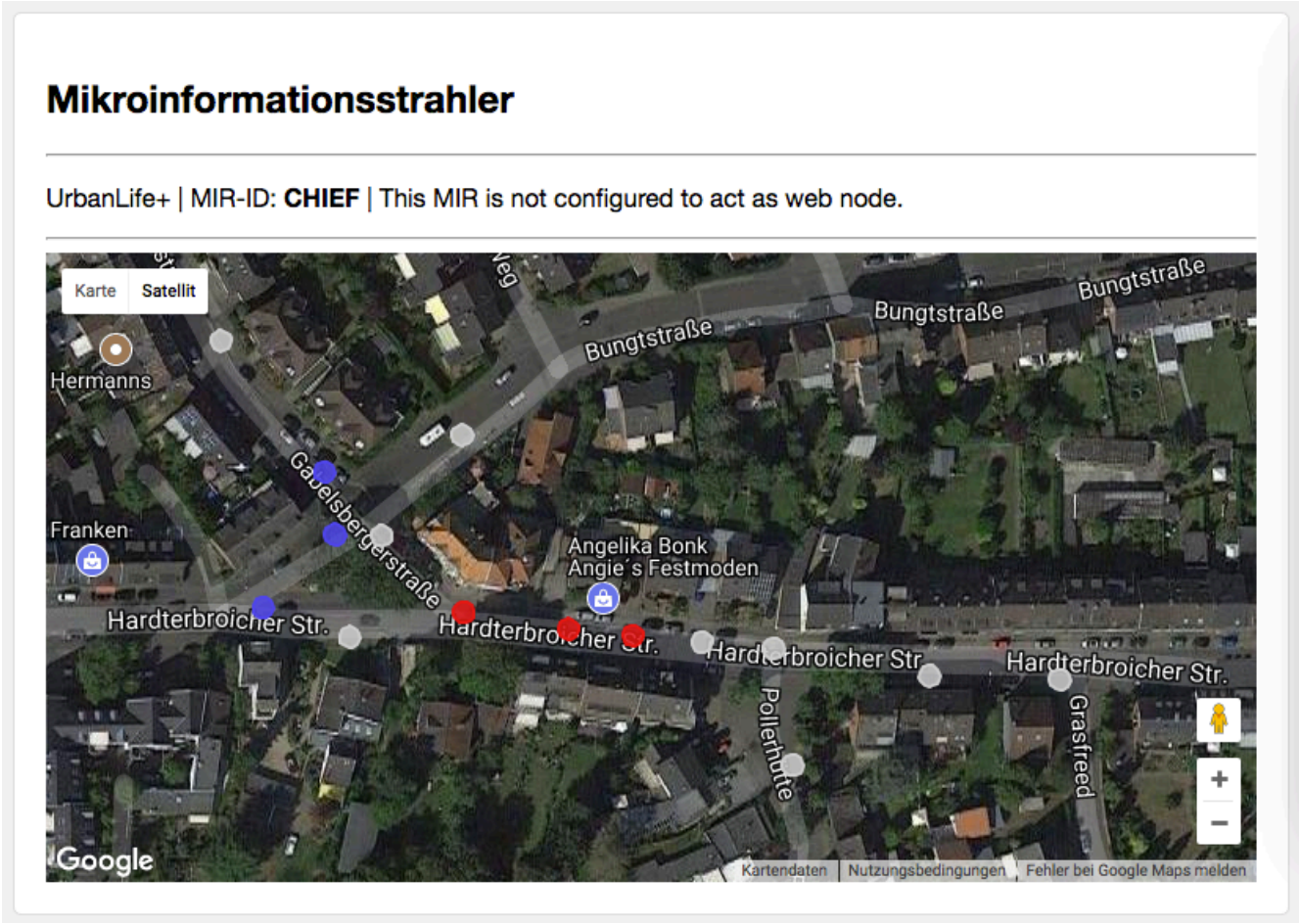

Abbildung 23 - Bildschirmfoto: Webstatusansicht für Tests

Abbildung 23 zeigt eine Ansicht der Website. Die via MQTT übertragenen Statusmeldungen enthalten die Koordinaten der Strahler, sowie die aktuell aktiven Farben. Die Ansicht integriert eine Google Maps Karte und zeichnet für jeden Strahler einen Punkt in der Karte ein. Weiße/graue Strahler sind inaktiv. Die blauen und roten Punkte zeigen jeweils eine aktive Routenführung und leuchten blau bzw. rot. Die MIS senden ihren Status etwa alle fünf Sekunden aus, entsprechend kann die Ansicht leicht verzögert sein.

Ereignisse können auf verschiedene Weise simuliert werden. Da der MQTT-Broker unter `iot.eclipse.org` unbeschränkten Zugriff erlaubt, können mittels JavaScript auch MQTT-Nachrichten von der Website aus veröffentlicht werden. Hierfür wurden einige Nachrichten- und Topic-Vorlagen in der Website hinterlegt, die nach Bedarf genutzt werden können. So lässt sich ein Smartphone simulieren, das sich Strahlern annähert. Die Kartenansicht aktualisiert sich nach solchen Events entsprechend.

5.10. Erkenntnisse

Die Prototypen wurden in kleinen Testaufbauten getestet, dabei konnte ihre Funktionalität gut illustriert werden. Die Arbeit mit Raspberry Pis hat sich als vorteilhaft erwiesen, da sie stark verbreitet sind und sie von diversen Ressourcen gut unterstützt werden. In realitätsnahen Umgebungen wurde jedoch keine Evaluation der MIS vorgenommen, deshalb kann insbesondere keine Aussage zur MCI-Akzeptanz o.ä. gemacht werden.

Die technische Implementierung hat dem Autor an verschiedenen Stellen gezeigt, wo das Architekturkonzept an Grenzen stößt. Wenn ein Anwendungsszenario von mehreren Ereignissen abhängt, so gestaltet es sich derzeit als schwierig, dieses zu implementieren. Reactoren reagieren derzeit lediglich auf einzelne Events. Kausale oder zeitliche Zusammenhänge zwischen mehreren Ereignissen können kaum genutzt werden. In Szenario 2.0 gibt es einen Abschnitt, in dem Margot einen MIS frägt, wo sich ihre Freundin Brigitte befindet. Der MIS antwortet, dass sich Brigitte an einem IS in der Nähe vom Markt-Haupteingang befindet. Die Antwort basiert darauf, dass Brigitte sich kürzlich dem Strahler am Eingang angenähert hat, doch zum Zeitpunkt als Margot nach ihr fragt ist dieses Ereignis längst verflogen. Hier benötigt es entweder eine zentral konsultierbare Instanz für solche Anfragen, oder die Möglichkeit vergangene Events zu cachen, um im Nachhinein Zusammenhänge zu erkennen und Schlüsse zu ziehen.

6. Zusammenfassung und Ausblick

In dieser Arbeit wurden neuartige Interaktionsparadigmen und Architekturkonzepte vorgestellt und angewandt. Es wurde gezeigt, dass mit einfachen Benutzerschnittstellen und simplen, kleinen Geräten komplexe Anwendungsszenarien gestaltbar sind. Dabei wurden von schlichten Lichtsignalen bis zu vielfältiger Sprachsteuerung verschiedenste Opportunitäten abgewogen. Im Ergebnis steht hier eine Szenario-Vision, in der Seniorinnen und Senioren durch intelligente, allgegenwärtige Mikroinformationsstrahler in Städten unterstützt werden.

Die technische Basis für dieses Szenario wurde durch die Entwicklung eines Architekturkonzepts und die prototypische Implementierung gelegt. In der Architektur wurden unterschiedliche Paradigmen angewandt, um den Anforderungen und Zielen des Szenarios gerecht zu werden. Dabei wird teils mit konventionellen Ansätzen gebrochen, so wird die im Internet of Things übliche Cloud-zentrische Applikation stellenweise um Konzepte des Fog Computing ergänzt, um Ziele des Datenschutzes, der Echtzeit und der Netzwerkauslastung zu erreichen. Die Anwendung setzt dabei nicht nur auf eine klassische serviceorientierte Architektur, sondern bietet auch die Flexibilität und Erweiterbarkeit einer eventgetriebenen Architektur.

In Konzept und Prototyp wurden aktuelle Technologien, wie iBeacons, MQTT, Sprachsteuerung und Continous Deployment gezeigt und dadurch ein Eindruck vermittelt, welche Potenziale diese für das Internet of Things offenbaren. Der Raspberry Pi bot eine solide Basis für die Prototypentwicklung.

iBeacons bieten im IoT-Umfeld vielversprechende Möglichkeiten impliziter Nutzerinteraktion. Ihre geringe Übertragungskapazität und ihre gerichtete Kommunikation machen dies aus, implizieren in Architekturüberlegungen jedoch auch Schwierigkeiten. Die sichere Übertragung eines größeren Datensatzes ist damit nicht ohne weiteres möglich.

MQTT hat sich in dieser Arbeit als vorteilhaft für die Kommunikation zwischen Komponenten erwiesen. Ein MQTT-Broker verteilt Daten an jene Komponenten, für die diese interessant sind Dabei hat sich die recht einfache Umsetzung des Publish/Subscribe-Modells auf Basis von Topics als ausreichend für diese Zwecke erwiesen. Bemerkenswert und wichtig für das Konzept ist die im MQTT-Standard enthaltene Fähigkeit der Authentifizierung und Autorisierung von Ressourcen. Ohne diese wäre es nicht möglich gewesen, die Architektur sicher zu gestalten.

Bei der Entwicklung des Prototyps wurden diverse Hilfsmittel und Werkzeuge erprobt und eingesetzt, die den Prozess teils erheblich vereinfacht und beschleunigt haben. Der Einsatz von Docker und Resin.io für ein zügiges und kontinuierliches Deployment hat die Aufwände zum Testen der MIS-Software stark reduziert. Die Arbeit, die in individuelle Werkzeuge zum Testen und Simulieren der Strahler gesteckt wurde, hat sich gelohnt, da zeitaufwendiges, manuelles Testen dadurch teils entfiel. Außerdem konnte durch die zusätzliche Webkomponente das Verhalten des Gesamtsystems besser verstanden werden.

Bei der Implementierung des Prototyps wurden erste Anzeichen der Grenzen des Architekturkonzepts erkannt. Die Orientierung an einer EDA hat zwar für Flexibilität gesorgt, doch zeigte sie Schwächen beim Umgang mit komplexen Ereignissachverhalten. Ein Reactor verarbeitet stets lediglich ein einzelnes Event. Dabei können kausale oder zeitliche Zusammenhänge zwischen mehreren Events nicht oder nur sehr umständlich erkannt und verarbeitet werden.

Mit Grund für die eventgetriebene und dezentrale, Fog-artige Architektur waren Datenschutzbedenken gegenüber der Cloud. Sensitive Daten sollten lediglich dort zur Verfügung stehen, wo tatsächlich notwendig. Dabei hat der Datenschutz insgesamt weitreichende Auswirkungen auf die Architektur. Dadurch, dass persönliche Informationen lediglich auf dem Smartphone eines Nutzers gespeichert werden, gibt es im Gesamtsystem keine zuverlässige Instanz, die für solche Daten angefragt werden kann. Hier ist besonders abzuwägen, ob es vertretbar ist, persönliche Daten, wie Bewegungsprofile und körperliche Eigenschaften, in der Cloud zu speichern.

Ausgehend vom Stand und von den Ergebnissen dieser Arbeit lassen sich einige weiterführende Fragen stellen und Ausblicke formulieren.

Die MCI der vorgestellten Mikroinformationsstrahler wurde hier nicht im Nachgang evaluiert. Durch Feldstudien oder Interviews mit Senioren könnten Erkenntnisse gesammelt werden, etwa wie die Akzeptanz gegenüber solchen Geräten ist, ob ihre Funktionsweise verstanden wird, ob sie die Zwecke des Szenarios erfüllen und ob eine Nutzerzufriedenheit erreicht wird. Auf dieser Basis ließe sich das Interaktionskonzept verbessern und ausbauen.

Während die vorgestellte Architektur einen insgesamt soliden Eindruck macht, gibt es dennoch Probleme, die noch gelöst werden müssen, um auch komplexere Anwendungen zu erlauben. Gemeint sind insbesondere Anwendungen, die sich aus einer Vielzahl von Ereignissen zusammensetzen. Im Ausblick lässt sich untersuchen, ob Methoden des Complex Event Processing hier eingesetzt werden können, um der Problematik zu entgegnen. In dem Zusammenhang ist auch interessant, inwieweit MQTT dies fördert.

Während dieses Konzept stellenweise bereits von intelligenten oder smarten Mikroinformationsstrahlern spricht, sind in dieser Arbeit noch keine spezifischen Bemühungen in dieser Richtung getroffen worden. In nachfolgenden Arbeiten könnte untersucht werden, welche Anwendungsmöglichkeiten künstliche Intelligenz oder Machine Learning in solchen urbanen Strahlern hervorbringen können.

Für den Kontext von UrbanLife+ außerdem interessant ist, ob und wie sich Interaktionsmöglichkeiten der Mikroinformationsstrahler auf interaktive Informationsbildschirme anwenden lassen. Insbesondere Sprachsteuerung könnte auch hier eine weitere Modalität bieten.

Szenario 2.0 zeigt bereits einige Beispiele, wie Sprachsteuerung in MIS zum Einsatz kommen kann. Im Prototyp wurde jedoch lediglich eine Sprachausgabe implementiert. Anders als Bildschirmeingaben erfordern Spracheingaben neben Natural Language Processing auch Methoden des Natural Language Understanding, um Eingaben auf tatsächliche Dienstschnittstellen umzulegen. Wie genau ein sprachgesteuerter MIS aufgebaut sein kann, könnte in einer Folgearbeit evaluiert werden.

Diese Bachelorarbeit liefert einen Beitrag zu MCI-Möglichkeiten mit sehr einfachen Licht- und Audiokomponenten und zu technischen Konzepten im Internet of Things. Dabei wurde das Thema nicht nur konzeptionell, sondern auch prototypisch behandelt. Die vorgestellte Fiktion von urbanen Szenarien mit Mikroinformationsstrahlern zeigen in Richtung einer größeren Vision, die bereits vor über 25 Jahren von Mark Weiser gezeichnet wurde: Ubiquitous Computing. Daher versteht sich diese Arbeit als wichtigen Zwischenschritt hin zu dieser Vision für die Städte von morgen.

Literaturverzeichnis

Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., & Ayyash, M. (2015). Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications. *IEEE Communications Surveys & Tutorials*, *17*(4), 2347–2376. https://doi.org/10.1109/COMST.2015.2444095

Atzori, L., Iera, A., & Morabito, G. (2010). The Internet of Things: A survey. *Computer Networks*, *54*(15), 2787–2805. https://doi.org/10.1016/j.comnet.2010.05.010

Ballagas, R., Rohs, M., & Sheridan, J. G. (2005). Sweep and point and shoot: phonecam-based interactions for large public displays. In *CHI '05 extended abstracts on Human factors in computing systems - CHI '05* (p. 1200). New York, New York, USA: ACM Press. https://doi.org/10.1145/1056808.1056876

Bonomi, F., Milito, R., Zhu, J., & Addepalli, S. (2012). Fog computing and its role in the internet of things. In *Proceedings of the first edition of the MCC workshop on Mobile cloud computing - MCC '12* (p. 13). New York, New York, USA: ACM Press. https://doi.org/10.1145/2342509.2342513

Botterman, M. (2009). *Internet of Things in 2020: Roadmap for the future*. Retrieved from http://cordis.europa.eu/pub/fp7/ict/docs/enet/iot-prague-workshop-report-vfinal-20090706_en.pdf

Brignull, H., & Rogers, Y. (2003). Enticing people to interact with large public displays in public spaces. *Human-Computer Interaction, INTERACT '03*, 17–24. https://doi.org/10.1.1.129.603

Bruns, R., & Dunkel, J. (2010). *Event-Driven Architecture*. Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-02439-9

Caceres, R., & Friday, A. (2012). Ubicomp Systems at 20: Progress, Opportunities, and Challenges. *IEEE Pervasive Computing*, *11*(1), 14–21. https://doi.org/10.1109/MPRV.2011.85

Campbell, J. P. (1997). Speaker recognition: a tutorial. *Proceedings of the IEEE*, *85*(9), 1437–1462. https://doi.org/10.1109/5.628714

Dalsgaard, P., & Halskov, K. (2010). Designing urban media façades: cases and challenges. In *Proceedings of the 28th international conference on Human factors in computing systems - CHI '10* (p. 2277). New York, New York, USA: ACM Press. https://doi.org/10.1145/1753326.1753670

Dekel, A., Simon, Y., Dar, H., Tarazi, E., Rabinowitz, O., & Sterman, Y. (2005). Adding Playful Interaction to Public Spaces (pp. 225–229). Springer, Berlin, Heidelberg. https://doi.org/10.1007/11590323_24

Dix, A., Finlay, J. E., Abowd, G. D., & Beale, R. (2004). *Human-computer interaction*. Pearson/Prentice-Hall. Retrieved from http://dl.acm.org/citation.cfm?id=1203012

Duden | Stadt-mö-bel | Rechtschreibung, Bedeutung, Definition. (n.d.). Retrieved May 10, 2017, from http://www.duden.de/node/831441/revisions/1298155/view

Fielding, R. T. (2000). *Architectural Styles and the Design of Network-based Software Architectures*. University of California, Irvine. Retrieved from https://www.ics.uci.edu/~fielding/pubs/dissertation/fielding_dissertation.pdf

Fischer, P. T., Gerlach, F., Acuna, J. G., Pollack, D., Schäfer, I., Trautmann, J., & Hornecker, E. (2014). Movable, Kick-/Flickable Light Fragments Eliciting Ad-hoc Interaction in Public Space. In *Proceedings of The International Symposium on Pervasive Displays - PerDis '14* (pp. 50–55). New York, New York, USA: ACM Press. https://doi.org/10.1145/2611009.2611027

Fischer, P. T., Hornecker, E., Umar, A., & Anusas, M. (2013). Urban HCI: PlazaPuck - An unowned, moveable, public interface. *Mensch & Computer 2013: Interaktive Vielfalt*, 293–296. Retrieved from http://dl.mensch-und-computer.de/handle/123456789/3312

Fischer, P. T., Hornecker, E., & Zoellner, C. (2013). SMSlingshot. In *Proceedings of the 7th International Conference on Tangible, Embedded and Embodied Interaction - TEI '13* (p. 9). New York, New York, USA: ACM Press. https://doi.org/10.1145/2460625.2460627

Fischer, P. T., Zollner, C., Hoffmann, T., Piatza, S., & Hornecker, E. (2013). Beyond information and utility: Transforming public spaces with media facades. *IEEE Computer Graphics and Applications*, *33*(2), 38–46. https://doi.org/10.1109/MCG.2012.126

Grudin, J. (2012). A Moving Target: The Evolution of Human-Computer-Interaction. In J. A. Jacko (Ed.), *The Human-Computer Interaction Handbook : fundamentals, evolving technologies, and emerging applications* (3rd ed., pp. xxvii–lxi). CRC Press. https://doi.org/10.1201/b11963-1

Gubbi, J., Buyya, R., Marusic, S., & Palaniswami, M. (2013). Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, *29*(7), 1645–1660. https://doi.org/10.1016/j.future.2013.01.010

Hinckley, K., & Wigdor, D. (2012). Input Technologies and Techniques. In J. A. Jacko (Ed.), *The Human-Computer Interaction Handbook : fundamentals, evolving technologies, and emerging applications* (3rd ed., pp. 95–132). CRC Press. https://doi.org/10.1201/b11963-9

Hoggan, E., & Brewster, S. (2012). Nonspeech Auditory and Crossmodal Output. In J. A. Jacko (Ed.), *The Human-Computer Interaction Handbook : fundamentals, evolving technologies, and emerging applications* (3rd ed., pp. 211–236). CRC Press. https://doi.org/10.1201/b11963-13

Ishii, H., & Ullmer, B. (1997). Tangible bits: towards seamless interfaces between people, bits and atoms. In *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '97* (pp. 234–241). New York, New York, USA: ACM Press. https://doi.org/10.1145/258549.258715

Jacob, R. J. K., Girouard, A., Hirshfield, L. M., Horn, M. S., Shaer, O., Solovey, E. T., & Zigelbaum, J. (2008). Reality-based interaction: a framework for post-WIMP interfaces. In *Proceeding of the twenty-sixth annual CHI conference on Human factors in computing systems - CHI '08* (p. 201). New York, New York, USA: ACM Press. https://doi.org/10.1145/1357054.1357089

Karat, C.-M., Lai, J., Stewart, O., & Yankelovich, N. (2012). Speech and Language Interfaces,

Applications, and Technologies. In J. A. Jacko (Ed.), *The Human-Computer Interaction Handbook : fundamentals, evolving technologies, and emerging applications* (3rd ed., pp. 367–386). CRC Press. https://doi.org/10.1201/b11963-20

Kostakos, V., & Ojala, T. (2013). Public Displays Invade Urban Spaces. *IEEE Pervasive Computing*, *12*(1), 8–13. https://doi.org/10.1109/MPRV.2013.15

Mathew, A. P., & Punnen, A. (2005). Using the environment as an interactive interface to motivate positive behavior change in a subway station. In *CHI '05 extended abstracts on Human factors in computing systems - CHI '05* (p. 1637). New York, New York, USA: ACM Press. https://doi.org/10.1145/1056808.1056985

Müllner, R., & Riener, A. (2011). An energy efficient pedestrian aware Smart Street Lighting system. *International Journal of Pervasive Computing and Communications*, *7*(2), 147–161. https://doi.org/10.1108/17427371111146437

Oviatt, S. (2012). Multimodal Interfaces. In J. A. Jacko (Ed.), *The Human-Computer Interaction Handbook : fundamentals, evolving technologies, and emerging applications* (3rd ed., pp. 405–430). CRC Press. https://doi.org/10.1201/b11963-22

Peltonen, P., Kurvinen, E., Salovaara, A., Jacucci, G., Ilmonen, T., Evans, J., … Saarikko, P. (2008). It's Mine, Don't Touch!: interactions at a large multi-touch display in a city centre. In *Proceeding of the twenty-sixth annual CHI conference on Human factors in computing systems - CHI '08* (p. 1285). New York, New York, USA: ACM Press. https://doi.org/10.1145/1357054.1357255

Peltonen, P., Salovaara, A., Jacucci, G., Ilmonen, T., Ardito, C., Saarikko, P., & Batra, V. (2007). Extending large-scale event participation with user-created mobile media on a public display. In *Proceedings of the 6th international conference on Mobile and ubiquitous multimedia - MUM '07* (pp. 131–138). New York, New York, USA: ACM Press. https://doi.org/10.1145/1329469.1329487

Poslad, S. (2009). *Ubiquitous Computing: Smart Devices, Environments and Interactions*. Chichester, UK: John Wiley & Sons, Ltd. https://doi.org/10.1002/9780470779446

Ryan, M. (2013). Bluetooth: With Low Energy comes Low Security. Retrieved from https://www.usenix.org/system/files/conference/woot13/woot13-ryan.pdf

Schroeter, R., & Ronald. (2012). Engaging new digital locals with interactive urban screens to collaboratively improve the city. In *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work - CSCW '12* (p. 227). New York, New York, USA: ACM Press. https://doi.org/10.1145/2145204.2145239

Seitinger, S., Taub, D. M., & Taylor, A. S. (2010). Light bodies: exploring interactions with responsive lights. In *Proceedings of the fourth international conference on Tangible, embedded, and embodied interaction - TEI '10* (pp. 113–120). New York, New York, USA: ACM Press. https://doi.org/10.1145/1709886.1709908

van Dam, A. (1997). Post-WIMP user interfaces. *Communications of the ACM*, *40*(2), 63–67. https://doi.org/10.1145/253671.253708

van der Bie, J., Visser, B., Matsari, J., Singh, M., van Hasselt, T., Koopman, J., & Kröse, B. (2016). Guiding the visually impaired through the environment with beacons. In

Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing Adjunct - UbiComp '16 (pp. 385–388). New York, New York, USA: ACM Press. https://doi.org/10.1145/2968219.2971387

Weiser, M. (1994). The world is not a desktop. *Interactions*, *1*(1), 7–8. https://doi.org/10.1145/174800.174801

Weiser, M. (1999). The computer for the 21st century. *ACM SIGMOBILE Mobile Computing and Communications Review*, *3*(3), 3–11. https://doi.org/10.1145/329124.329126

Zühlke, D. (2012). *Nutzergerechte Entwicklung von Mensch-Maschine-Systemen : Useware-Engineering für technische Systeme*. Springer.