

MSc Thesis

User Experience in Smart Environments

Design and Prototyping

June 28 2016

Written by
Tobias Uebbing BSc

Comittee
dr. Angelika Mader
dr. ir. Dennis Reidsma
ir. Hans Scholten
Till Schäfers MSc



UNIVERSITY OF TWENTE.

Abstract

Home automation has been a popular interest and desire of the broad public for decades. Recently those visions slowly have started to become a reality with so called “Smart Home” devices. Nevertheless, the market adoption of these devices does not meet the sales expectations of the manufacturers. Market surveys have shown that missing interoperability, intelligence and unpleasant User Experiences (UX) are potential causes. In-depth insights into the UX of such Smart Environments are scarce, since appropriate tools for the creation of rapid prototypes are missing. Therefore, UX designer and researcher have to invest high efforts into the creation of merely stable prototypes. This repeatedly leads to shortened user evaluations with unsatisfying results.

For this reason, specifications for and descriptions of components for a tool kit that supports, what we call, functional Experience Prototyping (functional ExP) were defined in the course of this project. According to our definition, functional ExP aims on creating functional prototypes that mediate the UX of an intended final product. It does not aim for technical finesse, superb scalability or stability. The generation of the tool specifications required two inputs: The desired UX that the toolkit needs to be able to generate and its potential impact on the common design and development process. Smart Environments have a broad application space and since UX is highly application-dependent the scope of this work was narrowed to a subdomain called Intelligent Living Environments (ILEs). General UX dimension were found based on the work of predecessors and an UX outline for ILEs including the user’s intention and context of use as well as system characteristics was established. Principles on the integration of user-centered agile design and development methods as well as lean and continuous development were gathered and merged. The resulting optimized method incorporates functional ExP and supports Continuous Interdisciplinary Prototyping (CIP). A prototyping environment that can be assembled of functional Experience - , Technical Prototypes and completely implemented components. This supports continuous development and provides an integrated testing environment for designers and engineers throughout the whole process. Based on the defined UX outline and optimized design and development method specifications for a functional ExP were derived. Subsequently these specifications were translated into actual toolkit components. As a final proof of concept some of these components were implemented and used to build use case prototypes. This final step proved that the specifications and component descriptions are a suitable foundation for the realization of a functional ExP toolkit for ILEs.

Acknowledgements

My deepest gratitude goes to to all people that supported and accompanied me during the execution of this graduation project as well as my whole master studies.

I would like to thanks my supervisors Angelika Mader, Dennis Reidsma and Hans Scholten for the feedback and guidance throughout this project. I am in particular grateful for the excellent advice and academic as well as emotional support by my daily supervisor Angelika Mader. She was not only attend to keep me focused and motivated, but also that I take care of myself during times of intense research and conception. Without our weekly reflective talks I would not have reached the results laying now in front of you.

In the same manner I would like to thank my wonderful boyfriend who supported me at least as much and always made me feel like all the effort will lead to a valuable piece of work. Additionally, this document became visually so appealing thanks to his noble graphical contribution. I also would like to thank my family for their steady support nevertheless which struggles occurred and for the sometimes undesired but needed distraction and relaxation.

Overall, I am thankful for my precious study years at the University of Twente that were enriched by my teachers, fellow students and friends. It would not have been the same without all of you.

At last I would like to thank the Deutsche Telekom AG and especially Matthias Böhmer, who helped me to develop the initial idea for this work, and Till Schäfers as my external supervisor throughout the project. Furthermore I would like to thank Thomas Kubitza from the University of Stuttgart for the cooperation, help and provided means. My appreciation also goes to the companies Nedap N.V. and eFocus for the possibility to interview them.

Contents

1	Introduction	1
1.1	Intelligent Living Environments	3
2	Background & Problem Motivation	6
2.1	Technical Background	6
2.2	Applications on the Market	15
2.3	User Experience Research	18
2.4	Experience Prototyping	20
2.5	Problem Statement	26
2.6	Functional Experience Prototyping	27
3	Desired UX	29
3.1	User Experience Dimensions	31
3.2	UX outline for ILEs	32
3.2.1	User's Internal State	33
3.2.2	Context of Use	35
3.2.3	System Characteristics	36
3.3	ILE Use Case Examples	42
4	Design & Development Method	46
4.1	Current Methods	46
4.2	An Optimized Method	53
5	A Functional Experience Prototyping Toolkit	61
5.1	Specifications	61
5.2	A Tool Framework	65
5.3	Implementation	72
6	Discussion & Conclusion	79
7	Future Work	81
	Bibliography	LXXXII
	Acronyms	LXXXVII
A	List of Identifiers	XCII
B	Evaluation Documentation	XCIV

List of Figures

1.1	Search interests over time.	1
1.2	Examples of Intelligent Applications.	2
1.3	Allocation of fields associated with Smart Environment and inter-relations.	5
1.4	Intelligent Living Environments (ILEs)	5
2.1	Identified IoT characteristics and derived technical challenges. . .	7
2.3	IoT architectures: (a) three-layer (b) service-oriented (c) five-layer	7
2.2	Technical elements of IoT as defined by Al-Fuqaha et al. [1]	8
2.4	Overview of standards relevant for IoT.	8
2.5	Overview of a selection of available development hardware. . . .	10
2.6	Overview of a selection of available development software. . . .	12
2.7	Illustration of Technical Prototyping	13
2.8	Multiple IoT Application Domain Spectra.	14
2.9	Overview of collected application examples.	16
2.10	Continued: Overview of collected application examples.	17
2.11	Illustration of Experience Prototyping	20
2.12	Overview of different types of prototyping activities.	28
3.1	The different scopes of User Experience (UX) design.	30
3.2	The three dimensions of UX.	32
3.3	General Intentions of Use for Intelligent Living Environments (ILE)s.	34
3.4	Overview of possible components of an ILE categorized into inputs, intelligence and outputs.	40
3.5	"Where did I leave ... my glasses?"	42
3.6	Remembrall in the Harry Potter Movie.	43
3.7	Ario Smart Lighting system over a day.	43
4.1	The Agile Manifesto as defined by Beck et al. [7]	47
4.2	One sprint in Scrum based on figure by Jongerius [29]	48
4.3	Illustration of optimized design methods for ILEs incorporating functional Experience Prototyping (functional ExP).	52
4.4	Presence of expertise within the team.	53
4.5	Evolution of the prototype through out the design & development process including functional ExP, technical and final components.	55
5.1	Overview over all components of the functional ExP toolkit.	67
5.2	Hardware of the meSch kit.	73
5.3	Device pool overview in the front end of the meSchHub	75
5.4	Grapiical User Interface (GUI) generated with Node-RED.	76

5.5 Screenshot of the several Application Program Interface (API) nodes connected to MQTT topics in Node-RED.	77
--	----

Chapter 1

Introduction

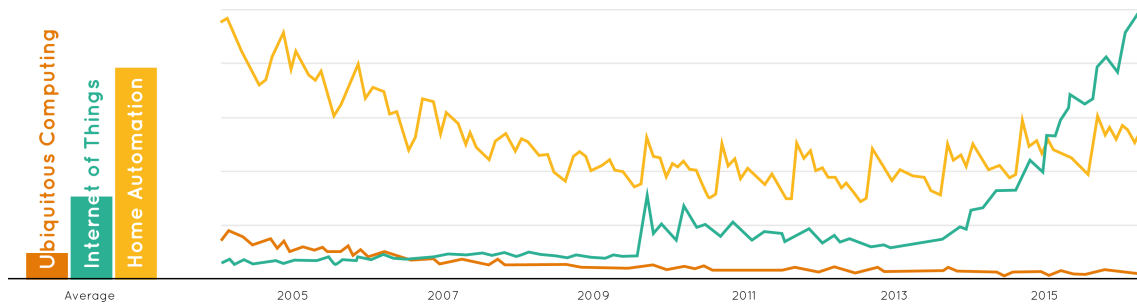


Figure 1.1: Search interests over time.

Source: [Google Trends](#)

The Internet of Things (IoT) is a term with continuously rising interest within research and industry since it was first coined in 1999 by Kevin Ashton [4]. The number of devices connected to the IoT is expected to reach 212 billion by the end of 2020 [17] and the annual global economic impact is estimated to be between \$2.7 and \$6.2 trillion by 2025 [43]. For this reason, leading Information Technology (IT) companies are making efforts to enter the IoT market. IBM, Google and Amazon are offering cloud services such as machine learning and IoT middleware for businesses. New hard- and software for IoT is constantly released for the development community by companies such as Samsung, Apple and others. Many of the big companies also offer end consumer products. Most popular application area in this market segment is the so called Smart Home. home automation has been a common interest for a while even before IoT was a well-known term as illustrated in figure 1.1. Current applications as for example smart thermostats and lighting systems can be either controlled via rules set by the user or manually from a remote location. But the adaption of these technologies by end consumers is still restrained and below expectations of manufacturers [12] [26].

The field has been mostly approached from an internet- and things-oriented perspective in the past [5, 44]. Extraction of meaning and semantic operability between systems has been mostly neglected as preparatory studies for this work have shown [5, 41, 44]. Intelligent products like the Nest

Learning Thermostat¹, the Ario Smart Lighting² system and Sense by Silklab³, a Smart Home hub, are already available on the market (see figure 1.2). They sense for example the presence of users and adapt their or the behavior of the environment accordingly. Nevertheless, those system are still isolated and keep gathered insights to themselves for now.



(a) Nest Thermostat



(b) Ario - Smart Lighting



(c) Sense by Silklab

Source: [Nest](https://nest.com/thermostat/meet-nest-thermostat/), [Ario](http://www.arioliving.com/), [Silklabs](http://www.silklabs.com/)

Figure 1.2: Examples of Intelligent Applications.

Connecting these systems shall lead to higher comfort, efficiency, security and empower users to manage their everyday life. But even if the technical challenges are solved still work is necessary to ensure a pleasant UX within these interconnected Smart Environments. However, first it needs to be clarified whether and how the UX of Smart Environments differs from the UX of stand-alone applications. UX design and research for Smart Environments and Context-Aware Applications and similar require high effort in terms of prototyping and user testing [55]. Questionable is to which extent design methods for stand-alone applications are suitable for UX design within such Smart Environments.

This work investigates how work of designers and researcher focusing on UX within Smart Environments can be facilitated. The focus will be sharpened to the area of ILEs on reasons explained within section 1.1. Chapter 2 provides a knowledge base in technical aspects, current applications on the market and UX research. Furthermore the chapter introduces the term Experience Prototyping (ExP). Concluded is the chapter by summarizing the problem motivation and formulating a problem statement. Before the problem can be tackled chapter 3 defines the desired UX of ILEs. This generic outline includes three UX dimension aspects. Chapter 4 presents current common design & development methods and proposes a method that is optimized for the appli-

¹<https://nest.com/thermostat/meet-nest-thermostat/>

²<http://www.arioliving.com/>

³<http://www.silklabs.com/>

cation of functional ExP. In chapter 5 we derive tool specifications from prior gathered knowledge and translate those to actual components of a software toolkit. Furthermore this chapter describes how an initial set of tool components is implemented. Finalized is the chapter by a proof of concept of the toolkit executed by the implementation of real world ILE use cases to validate the applicability of the tools. Afterwards, the findings and outcomes of this work are discussed and conclusions are drawn in chapter 6. Concluded is this work with a prospect on future efforts to continue the work on facilitating the design and prototyping of UX within ILEs and Smart Environments in chapter 7.

This work is a graduation project from the Master studies in Human Media Interaction at the University of Twente, Netherlands. It is carried out for the Products & Innovation department of the telecommunication provider Deutsche Telekom AG situated in Germany.

Final outcomes of this project are:

1. Generic outlines for the UX of ILEs.
2. A proposal for an improved design and development method for ILEs evaluated by experts.
3. Specification for a tool framework that supports this process and their translation to specific software components.
4. A collection of use cases inspired by previously defined UX outlines as well as current existing application in the market.
5. Implementation of an initial set of these tools and a proof of concept in form of an implementation of use cases using the toolkit.

1.1 Intelligent Living Environments

There are many names for the same phenomenon existing that highlight different aspects: Internet of Things, Physcial Computing, Smart Environments, Ambient Intelligence (AmI) or Ubiquitous Computing (UbiComp). UbiComp, first introduced by Mark Weiser [56], is the paradigm to embed technology into the background of everyday life [20] and claims to be the field of origin of related terms. Kuniavsky [35] claims this area of fields never had a structure and implies every attempt to structure it would lead to different results. While working on this project and reading about all different terms it became clear all fields are related to some degree and also share intersections. Figure 1.3 shows our vision on the allocation of the terms and their relation to each other. The outer circle contains the three approaches to the field as defined by Atzori et al. [5]: Internet-, things- and semantic-oriented approach. Aligned with these approaches on the next inner circle five main research fields were identified:

- Ubiquitous Computing (UbiComp) - embedding technology into everyday life
- Physcial Computing - connecting computing to the physical world by sensors and actuators
- Cloud Computing - granting access to cloud services for computing and communication in-between systems indirectly and asynchronously
- Big Data - managing, storing and handling the significant high amount of data that is expected to be produced
- Artificial Intelligence (AI) - extracting meaning from data and act on it intelligently

In the center of figure 1.3 the multidisciplinary fields were allocated in between their corresponding main fields. Most popular term right now is the Internet of Things (IoT). However, in our point of view IoT does not incorporate the extraction of semantic meaning and resulting intelligence. The terms that match our perception of the general phenomenon most are Smart Environments and Aml, where Aml forms more of an enabling base for Smart Environments.

For this reason, the focus of this work was laid on Smart Environments. Preparatory studies have shown that Smart Environments includes and intersects with many application domains such as smart cities, environmental monitoring, smart home, agriculture, markets and industry. It is evident that the UX of an application or service unfortunately depends highly on the user, his Intention of Use, Context of Use and application specific aspects [10, 22, 38]. The attempt to work on the UX of something as broad and general as UbiComp or Smart Environment would potentially exceed the scope of this project. Consequentially, the focus is narrowed on an application domain that we call Intelligent Living Environments (ILEs). ILEs include all aspects that are incorporated in a common everyday private live of a person. Therefore ILEs include terms such as Smart Home and Ambient Assisted Living (AAL) but also extents to private transportation and contact points with external services such as businesses. Figure 1.4 illustrates our point of view on the scope of ILEs.

At last in this introduction two minor remarks concerning the reading of this work. First, even though the terms (ILE, IoT, UbiComp, etc.) are so closely related we preserved the original word choice when citing work of predecessors. Treating them as different aspects of the same phenomenon potentially facilitates the reading process of this work. Second, a lot of different terms and aspects are defined and used throughout this work. For this reason, we introduced identifiers such as [EX.1] to uniquely mark them. A complete List of Identifiers is available in appendix A.

The following chapter provides background information on the state of the art in applications, technical and UX prototyping and design methods as well as in UX research.

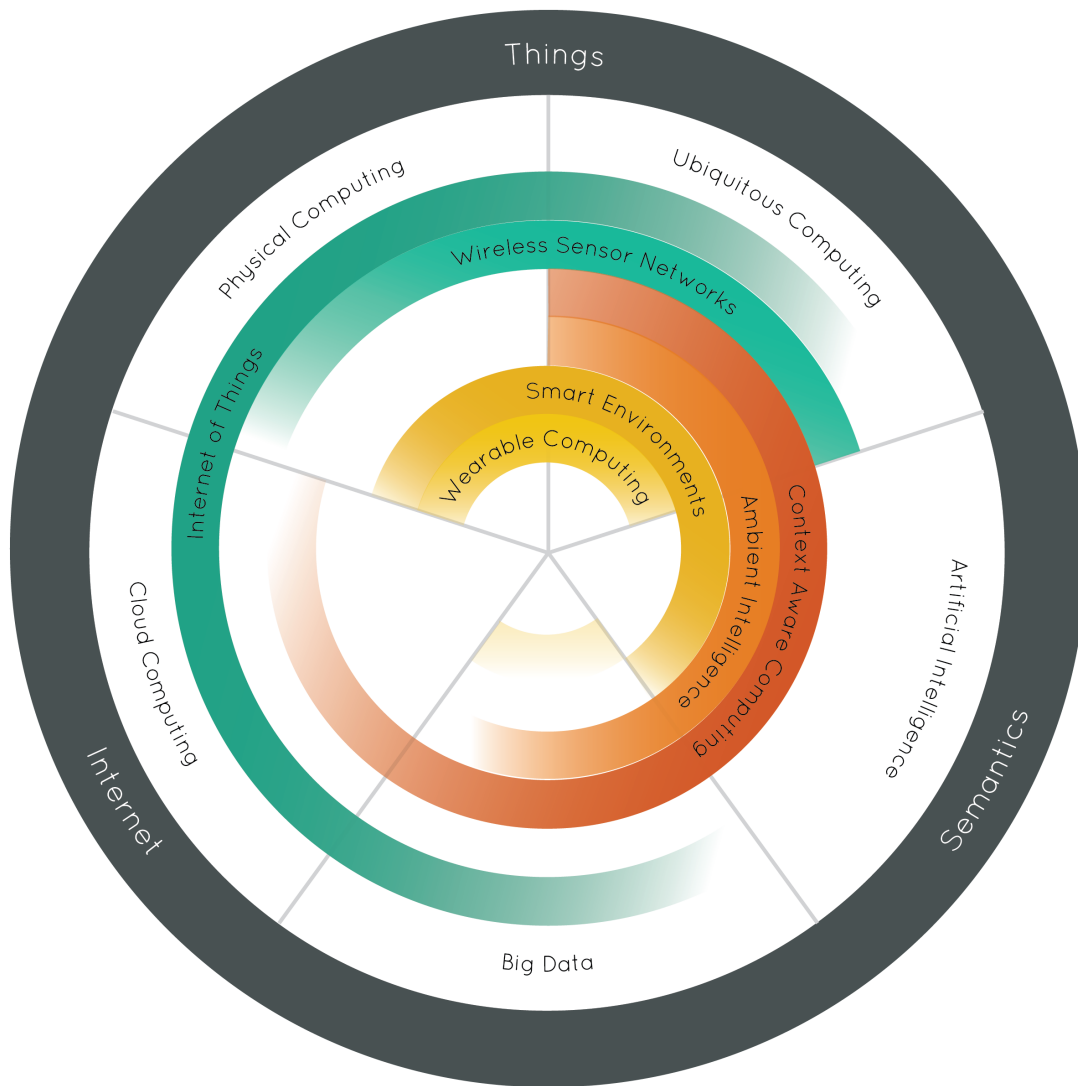


Figure 1.3: Allocation of fields associated with Smart Environment and interrelations.

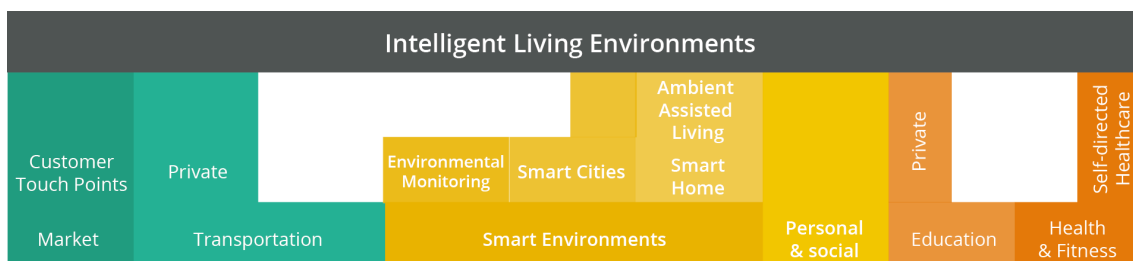


Figure 1.4: Intelligent Living Environments (ILEs)

Chapter 2

Background & Problem Motivation

This chapter forms a knowledge foundation for the further course. Additionally it provides the motivation for and specifies the problem tackled in this work in more detail. At first technical background knowledge relevant for the field concerning system characteristics, standards and hard- and software is given. Afterwards, existing applications in the market are briefly presented and examined with focus on matching the demands of the end consumers. Followed by a summary on the state of the art within User Experience (UX) research of Ubiquitous Computing (UbiComp) environments. The subsequent section introduces the method of Experience Prototyping and elaborates on supporting prototyping methods and their suitability for the design of Intelligent Living Environments (ILE)s. The first part of this chapter summarizes the results of our preparatory studies for this work on the subject of advances in applications and development for Internet of Things (IoT) and related fields.

2.1 Technical Background

Information Technology (IT) experienced significant advances in the last decade. Computing power steadily increased dramatically while the size of components and their cost decreased with the same factor. Shifting Mark Weiser's [56] vision more and more into reality. The introduction of wireless technologies such as RFID, NFC, WiFi and BLE made UbiComp systems such as Wireless Sensor Networks (WSN) and IoT possible. In the following information on the characteristics of IoT and related systems, important standards, hard- and software, and prototyping tools are explained.

System Characteristics

Miorandi et al. [44] defines the three main system-level characteristics within IoT so that 'anything' communicates, 'anything' is identified and 'anything' interacts. From these main more specific characteristics can be derived. The 8 characteristics and resulting technical challenges as identified by predecessors [1, 41, 44, 50] are listed in figure 2.1. The characteristics and challenges can be projected on ILEs due to the close relations between Smart Environment and IoT. For more details please consult the preparatory study for this work.

IoT applications are built-up from various technical components. Al-Fuqaha et al. [1] composed an extensive survey on the technical advancement within

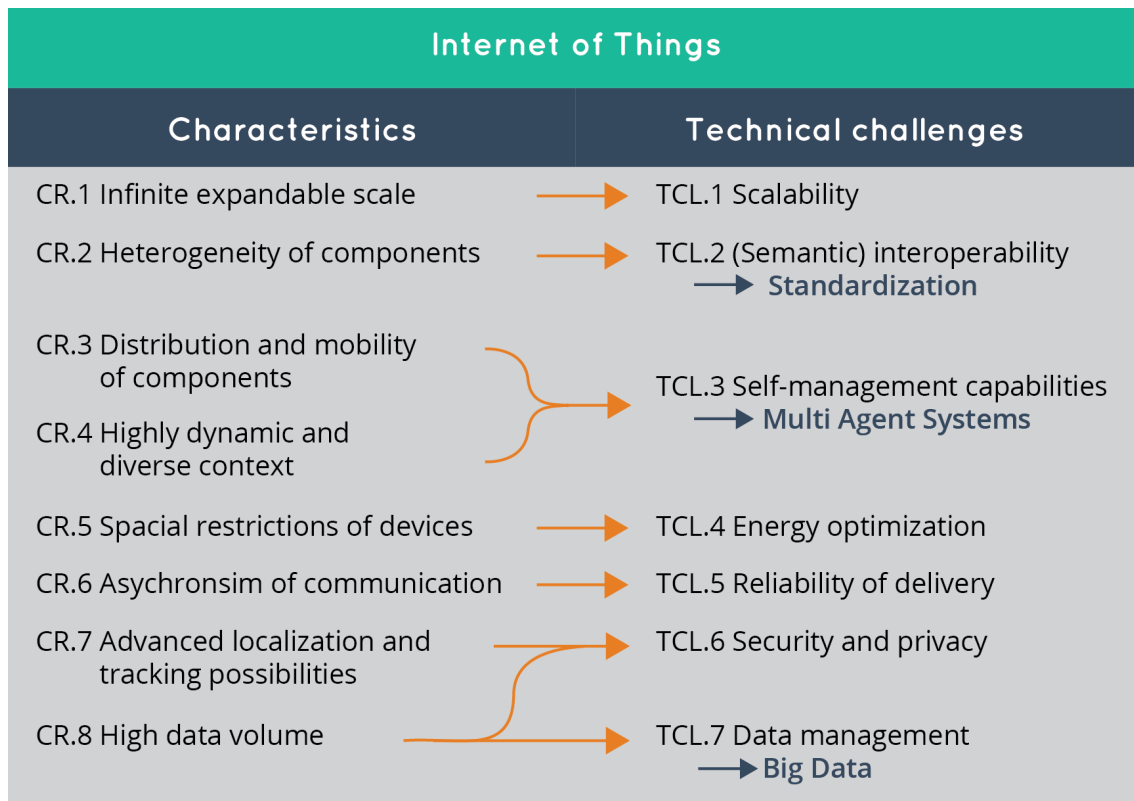


Figure 2.1: Identified IoT characteristics and derived technical challenges.

the field. They discuss architecture, building blocks, common standards, challenges and some application use-cases. The technical building blocks of IoT identified by them are shown in figure . When designing and developing IoT, the challenges identified previously such as scalability, heterogeneity and interoperability should be taken into consideration [41].

According to Al-Fuqaha et al. [1] a flexible layered architecture is therefore a critical necessity to connect a theoretical infinite amount of heterogeneous objects. However, “the ever increasing number of proposed architectures has not yet converged to a reference model” [1]. Figure 2.3 illustrates three different generic approaches: a basic three-layer architecture [31, 59, 60], a service-oriented architecture (SoA) [1, 41, 44] and a five-layered architecture [31, 59, 60]. In the further course of this subsection the five-layered architecture will be used as reference to position different standards and technologies.

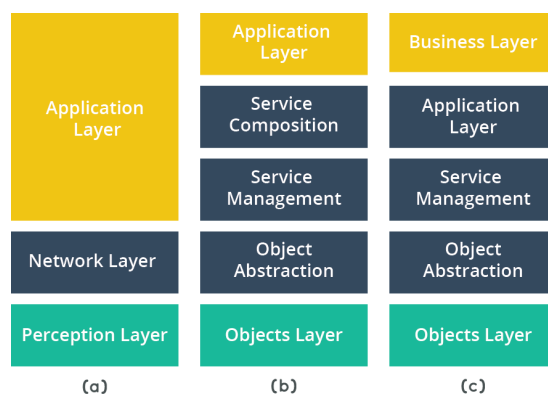


Figure 2.3: IoT architectures: (a) three-layer (b) service-oriented (c) five-layer

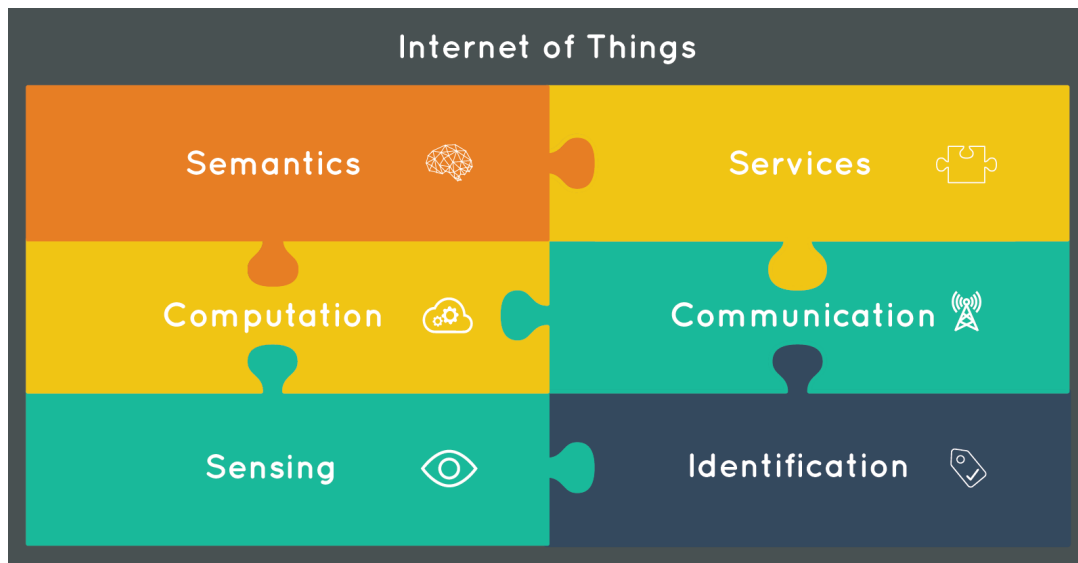


Figure 2.2: Technical elements of IoT as defined by Al-Fuqaha et al. [1]

Standards

Standardization is a key success factor for IoT to solve the challenge of heterogeneity [41, 57]. In the following common standards for different component and architectural layers are briefly introduced.

For identification & communication Infrastructure Protocols are needed to ensure interoperability within the Objects and Object Abstraction Layer. Protocols for identification are divided into identifying entities by an identifier such as EPC as implemented within RFID and addressing them by IPv4 and IPv6. 6LoWPAN provides a compressions mechanism for IPv6 to make IPv6 addressing possible on low power wireless networks. Most communication standards are well-known such as WiFi, BLE, RFID and NFC. Other standards as e.g. LoRaWAN and LiFi are currently emerging.

Application Protocols are jointly used by hosts in a network to com-

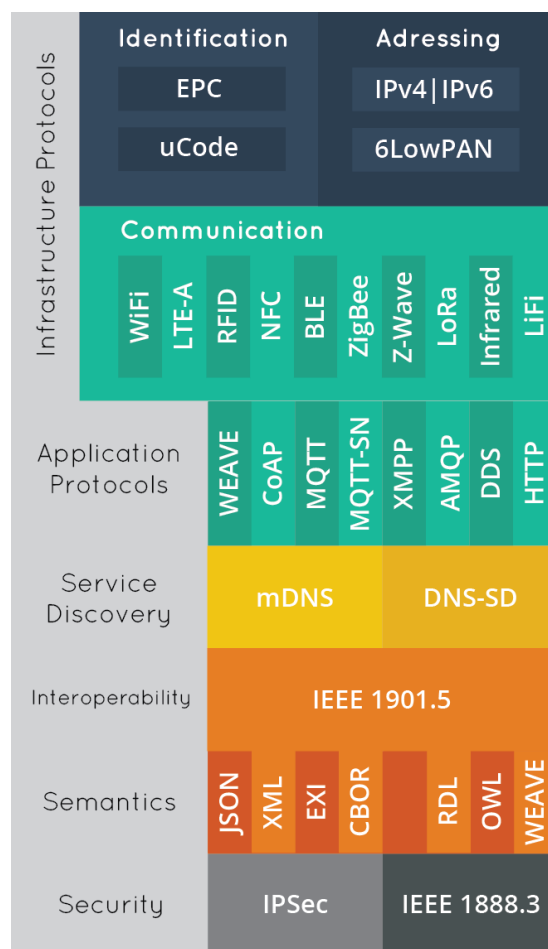


Figure 2.4: Overview of standards relevant for IoT.

municate to each other. Standards that are currently commonly used in IoT and similar systems are CoAP and MQTT. CoAP is based on REST on top of HTTP. UDP is the underlying transfer protocol and for identification URIs are used. CoAP incorporates mechanism that fix unreliability of message delivery within UDP. MQTT is built on top of TCP as transfer protocol. It implements a publish/subscribe pattern that allows one-to-one, one-to-many, and many-to-many relations. An MQTT implementation consists out of three components: publishers, subscribers, and one broker. Clients can be subscriber as well as publisher. A new emerging communication protocol is Weave but it is still in development and in invite-only Beta phase.

(Semantic) Interoperability “[T]here is a need for interoperation of underlying technologies” [1] due to the amount of different communication standards available and general heterogeneity present. The standard IEEE 1905.1 was introduced to converge technological heterogeneity. Other standardization efforts also support these endeavors. Unfortunately semantic interoperability has been mostly neglected. Though there are formal standards for the structure of information like JSON, EXI(XML), RDF and OWL there are no overarching agreements to make them applicable within IoT devices and services.

Figure 2.4 provides an extended overview of standards important for IoT and similar systems.

Hard- & Software

The hard- and software market for the development of IoT and similar application is fast emerging. In this section current new releases are presented.

Hardware -wise the two main segments are micro-controller boards and single-board computer. Most well-known micro-controller prototyping platform is Arduino¹, respectively Genuino in Europe since 2015. For this reason, many other platforms are compatible with Arduino hardware and the associated Integrated Development Environment (IDE). Many new micro-controller boards have connectivity modules included such as for WiFi, BLE or LoRaWAN. Single-board computers became popular with introduction of the Raspberry Pi². The low-budget computer was primarily meant to enable access to computation and the internet in education and developing countries. The newest version 3 Model B also includes a module for WiFi and BLE connectivity. Competitors such as the UDoo Neo³, aimed on IoT development, also already incorporates sensors on-board. Figure 2.5 provides an more extensive overview on current available hardware.

¹<http://www.arduino.cc/>

²<https://www.raspberrypi.org/>

³<http://www.udoo.org/udoo-neo/>

Manufacturer	Board	System on a Chip	Clockspeed	SRAM DRAM	Storage Memory	Supply - Operation Voltage	Characteristics	Price
Arduino	MKR1000	ATSAMW25	33 Mhz	32 KB	256 KB	5V 3,3V	low power Wifi	30,99€
Arduino	Yùn	ATmega 32u4 Linino AR 9331	16 MHz	2,5 KB	32 KB 16 MB	5V 5V 5V 3,3V	Ethernet Wifi	52,00€
Particle	Photon	STM32F205	120 MHz	128 KB	1 MB	3,3V 3,3V	WiFi	19,00\$
Particle	Electron	STM32F205	120 MHz	128 KB	1 MB	12V 3,3V	Cellular (2G 3G)	49,00\$
Punchthrough	Lightblue Bean	ATmega328p CC254X	8 MHz	2 KB	32 KB	3V	BLE, RGB LED, Accelerometer	30,00\$
PyCom	LoPy	dual core	2 x 160 MHz	256 KB	1 MB	5,5V 3,3V	LoRa, Wifi, BLE	29,95€
mbientlab	MetaWear C	ARM Cortex - MD	2,33 MHz	16 MB	1 MB	3,3 V	BLE, various Sensors	50,00\$
Intel	Galileo	Intel Quark SoC X1000	400 MHz	16 KB	up to 2 GB	7-15V 3,3V	Ethernet Arduino compatible	65,00€
Samsung	Artik 1	dual core	250 MHz 80 MHz	1 MB	4 MB	5V ?	BLE	unknown
Samsung	Artik 5	dual ARM Cortex A7	2 x 1 GHz	512 MB	4 GB	3,4-5V ?	Wifi, BT BLE, ZigBee	99,99\$
Samsung	Artik 10	4 x ARM Cortex 15 4x ARM Cortex 7	4 x 1,5 GHz 4 x 1,3 GHz	2 GB	16 GB	3,4-5V ?	Wifi, BT BLE, ZigBee	149,99\$
Raspberry	Pi 3 Model B	quad core ARM Cortex A53	4 x 1,2 GHz	1 GB	up to 128 GB	5V 3,3V	Ethernet Wifi, BLE	35,00\$
UDOO	Neo	ARM Cortex A9 ARM Cortex M4	1 GHz 200 MHz	1 GB	up to 128 GB	5V 3,3V	Ethernet, Wifi, BLE, Sensors	64,90\$
Intel	Edison	Intel Atom Intel Quark	500 MHz 100 MHz	1 GB	4 GB	3,3-4,5V 3,3V	BLE	49,99\$

Figure 2.5: Overview of a selection of available development hardware.

Software and Services for IoT development are available in various forms. This paragraph will shortly introduce relevant operating systems (OS)s, middleware and cloud services. For single-board computers different operating systems are available. Two new OSs are Windows 10 IoT Core⁴ and Brillo⁵ by Google. Brillo claims to provide essential functionalities for IoT devices such as over-the-air updates. middleware compensates for asynchronism and ensures reliability of delivery within a system. It can either run within a local network or remotely on a server as cloud service. Example for middleware that can run on a local scope are Node-RED⁶, Eclipse SmartHome⁷ or the meSchup server developed during the meSch project [34]. Other middleware is offered in form of a cloud service like AWS IoT⁸ or IBM Watson IoT⁹. Other big IT companies that offer cloud services including computing, storage and machine learning capabilities are Google¹⁰ and Microsoft¹¹. Figure 2.6 provides an more extensive overview on current available software and services.

⁴<https://developer.microsoft.com/en-us/windows/iot>

⁵<https://developers.google.com/brillo/>

⁶<http://nodered.org/>

⁷<http://www.eclipse.org/smarthome/>

⁸<https://aws.amazon.com/iot/>

⁹<http://www.ibm.com/internet-of-things/>

¹⁰<https://cloud.google.com/>

¹¹<https://azure.microsoft.com/>

Operating Systems	Windows 10 IoT Core	Raspbian	Google Brillo (Android)	Contiki OS
	LiteOS	TinyOS	Riot OS	ARM mbed OS
Middleware	IFTTT	meschKit	Oracle Fusion	Kaa
	IBM Node-Red			AWS IoT
Cloud Platforms	IBM Bluemix	Particle Cloud	Google Cloud Platform	Amazon Web Services
	MS Azure	LittleBits Cloud	Samsung SmartThings	RealTime.io

Figure 2.6: Overview of a selection of available development software.

Technical Prototyping

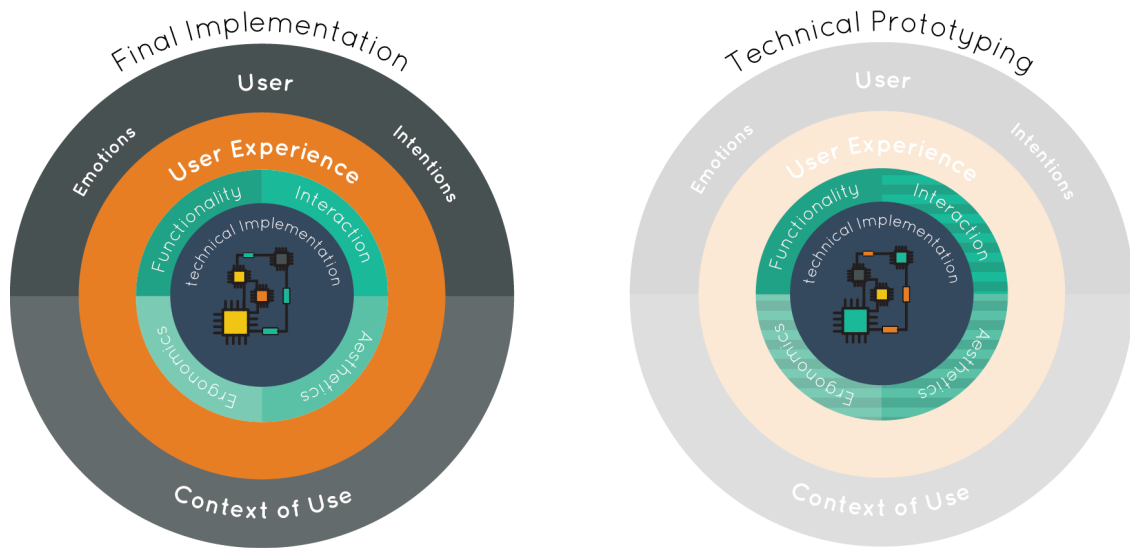


Figure 2.7: Illustration of Technical Prototyping

We understand Technical Prototyping as following:

Technical Prototyping is the prototypical implementation of a system towards a technical realization and deployment of a concept. This activity can have different fidelity levels as aim from exploration and feasibility studies to final implementation.

The advancements in available off-the-shelf development hard- and software facilitates faster and more cost effective prototyping of IoT applications. New development platforms and boards released also simplify prototyping activities significantly. This is also facilitated by accessories for hardware platforms like Arduino in form of easily attachable and accessible shields. Many other platform also offer Arduino compatibility in hard- and software as well as IDE due to its popularity. Other development platforms such as offered by Particle or Google Brillo promise a easy transition and thus scalability from prototyping over development to deployment.



Figure 2.8: Multiple IoT Application Domain Spectra.

2.2 Applications on the Market

One result of the preparatory study was to define an updated application domain spectrum based on similar spectra defined by predecessors (see figure 2.8) [1, 5, 41, 44, 57]. The proposed spectrum with 8 different domains is shown in the bottom of figure 2.8. The domains were defined with existing and potential future applications in mind.

Second step was the collection of current application examples. Figure 2.9 shows an overview of the applications collected. As elaborated in section 1.1 this work concentrates on ILEs which mostly incorporate the domains of Smart Environments, Personal & Social, Education and Health & Fitness (see also figure 1.4). For this reason, most examples collected within the preparatory study fall under these domains.

Typical current end consumer products are retro-fit Smart Home solutions for controlling heating, lighting and the power supply of devices, and monitoring energy consumption. Smart Home hubs such as Smart Things [EX.23] or Homey by Athom [EX.24] are often a part of such ecosystems. They communicate to devices using various communication standards, provide internet access to those and execute conditional rules which emulate the smart behavior of the environment. More intelligent products like the Nest Thermostat [EX.11.1] that adapt to presence, preference and daily routine of the user by themselves are available, too. Other products like the Cognitoys [EX.36] use cloud services to access speech recognition and synthesis capabilities as well as knowledge sources. Connected environments are also used to make museum exhibitions more interactive. The meSch project [EX.37] funded by the European Union is aiming to facilitate the use of such technologies within museums.

The critical aspects identified within this collection include missing interoperability of products and the lack of (sensor) infrastructure in buildings and environments. Improvements within these aspects would potentially lead to higher consumer interest in IoT products. End consumer indicated in a survey that missing interoperability is a major critic point of Smart Home products [25]. For this survey Icontrol interviewed 1600 north american consumers in spring 2015 on the topic of Smart Home adoption. The results show that the current products are not meeting actual desires of end consumers.

In many current systems the user can either remote control functionalities via the internet and/or predefine rules and preferences that are triggered by specified sensor inputs or events. 60% of the participants of the survey indicated that they would rather prefer devices that use sensors, data and analytics to act on their own. Next best scenario from the consumer's point of view is the interaction via voice-control or text messages (chat bots). Both of these characteristics are already available in some products. The before mentioned Nest Learning Thermostat [EX.11.1] and the Ario Smart Lighting system learn about their user and adapt accordingly. Amazon's Alexa [EX.27] and Siri [EX.25] make it possible to control devices using speech input. The problem with those systems is that they often operate only on their own isolated platform. It necessary that those system can communicate and understand each

Application Domain		Identifier	Name - Description	Company Author
Industry		EX.01	Fire protection system	Generic
		EX.02	Mining production monitoring & safety	Generic
		EX.03	Collaborative warehouse system	Reaidy et al. [47]
Agriculture		EX.04	Health & growth monitoring of plants & animals	Generic
Market		EX.05	Electronics shelf labels (ESL)	Generic
		EX.06	Self-managing & restocking inventory	Generic
		EX.07	Electronic price tags (RFID)	Generic
Transportation		EX.08	Central automated coordination of traffic flows	Generic
		EX.09	Self-driving car	Generic
		EX.09.1	by	Google
		EX.09.2	by	Tesla Motors
		EX.09.3	by	Audi
		EX.09.4	by	General Motors
Smart Environments	Lighting	EX.10	Connected lighting solutions	Generic
		EX.10.1	Hue	Philips
		EX.10.2	LIFX	LIFX
		EX.10.3	Lightify	Osram
	Heating	EX.11	Smart thermostats	Generic
		EX.11.1	Nest Learning Thermostat	Nest
		EX.11.2	by	netatmo
		EX.11.3	tado	tado
	Sensing	EX.12	Connected weather station	netatmo
		EX.12.1	Complimentary wind gauge	netatmo
		EX.12.2	Complimentary rain gauge	netatmo
		EX.13	Connected Camera Sensors	Generic
		EX.13.1	Welcome - Indoor camera with motion detection and face recognition	netatmo
		EX.13.2	Presence - Outdoor camera & light with detection of humans, animals & cars	netatmo
		EX.13.3	Nest Cam - Indoor camera with motion detection	Nest
		EX.14	Nest Protect - Connected smoke & carbon monoxide detector	Nest
		EX.15	SleepSense - Connected sleep monitoring device	Samsung
		EX.16	PlantSitter - Plant monitoring for private houses	MiGrow Inc
		EX.17	oombrella - Smart umbrella	wezzoo
		EX.18	Dash Replenishment Service - Easy and/or automated restocking of goods at home via Amazon deliveries	Amazon
		EX.18.1	Dash Button - ordering staple goods at Amazon with one button press	Amazon
		EX.18.2	Infinity Smart Water Pitcher - order new filters automatically	Britta
	Security	EX.19	Quantified Art - Artwork that indicates CO ₂ levels within a room	netatmo
		EX.20	Smart locks	Generic
		EX.20.1	August	August Home
		EX.20.2	Bolt	Lockitron
	Hubs	EX.21	Algorithm that balances comfort and energy efficiency	Anvari-Moghaddam et al. [3]
		EX.22	Agents that resolve conflicts in Smart Environments by negotiation	Jiang and Fei [27]
		EX.23	SmartThings - Smart Home Hub	Samsung
		EX.24	Homey - Smart Home Hub with speech interface	Athom
	IPAs	EX.25	Siri	Apple
		EX.26	Google Now	Google
		EX.27	Alexa	Amazon
		EX.27.1	Amazon Echo, Echo Dot and Echo Tap - necessary Alexa hardware	Amazon
		EX.27	Sirius - Open-source framework for IPAs	Clarity Lab

Figure 2.9: Overview of collected application examples.

Application Domain	Identifier	Name - Description	Company Author
Social & Personal	EX.29	Smart watches	<i>Generic</i>
	EX.29.1	Gear 2	Samsung
	EX.29.2	Moto 360	Motorola
	EX.29.3	Calendar Watch	What?
	EX.29.4	Apple watch	Apple
	EX.30	Connected jewelry with notification capabilities	Vinaya
	EX.31	Mediating the presence of remotely living close ones	<i>Generic</i>
	EX.31.1	Good Night Lamp - Set of connected and synched lamps	The Good Night Lamp
	EX.31.2	Avakais - Connected interactive wooden toys for children	Vai Kai
	EX.31.3	En Kompis - Connected hospital robot buddy for children	Linus Sundblad
	EX.31.4	Ambird - Ambient messaging system	Jespersen et al. [26]
	EX.31.5	Kettle Mate - connected electric water boiler and tea box	Brereton et al. [7]
	EX.31.6	Ghosting - Simulating presence of remote person with Smart Home	Clark and Dutta [12]
Health & Fitness	EX.32	Ambient Assisted Living (AAL)	<i>Generic</i>
	EX.32.1	IoT architecture for long-term care	Coelho et al. [13]
	EX.32.2	Smart water bottle	Lee et al. [38]
	EX.33	Tangible out of body experience (TOBE)	Gervais et al. [18]
	EX.34	Stress Ball	Simone Schramm
	EX.35	Fitness wristband	<i>Generic</i>
	EX.35.1	by	Fitbit
Education	EX.32.2	by	Garmin
	EX.36	Cognitoys - connected educational toys for children	CogniToys
	EX.37	Facilitating creation of interactive exhibitions spaces in museums	meSch Project

Figure 2.10: Continued: Overview of collected application examples.

other to create intelligent environments. Therefore (semantic) interoperability is key to enable the sharing and exchange of data.

The survey by iControl Networks [25] also identified another key success factor for the adoption of Smart Home systems. Surveyed consumers agreed that the UX and especially the ease-of use is more important than technical innovation. Väänänen-Vainio-Mattila et al. [55] agree that the consideration of the UX in the design and development process of intelligent environments from the very beginning is crucial for their success. As identified within our preparatory study it is necessary for the future that systems recognize and understand their users and act appropriately on their own. However, it does need to be ensured that the user also feels comfortable being surrounded by systems that autonomously adapt their behavior and manipulate the environment.

The following section will introduce the term of User Experience (UX) in more detail and will elaborate on the state of the art in research on UX for IoT, UbiComp and other related fields.

2.3 User Experience Research

So as seen in the previous section IoT and related fields are maturing in terms of technology and started to enter the market [55]. Human Computer Interaction (HCI) and UX research has been a point of interest for many scholars within the field [54, 55]. “UX as a field seeks to offer a systematic approach to design and analysis of the user’s holistic experiences with the technology” [55]. Don Norman stated in 2000 that he created the term UX because he thought the term human interface and usability were too narrow and he wanted to cover all aspects of a person’s experience with a system [23]. Since the acceptance of a system “depends on how the user experiences them in real context”, well-executed UX Design has become a quality attribute and important success factor of any technology [55].

However, the concept of UX is still unclear and vague for many researchers and designers [15] and many different definitions exist¹². The definitions by Alben [2] and Kuniavsky [35] match our understanding of UX closest:

“All the aspects of how people use an interactive product: the way it feels in their hands, how well they understand how it works, how they feel about it while they’re using it, how well it serves their purposes, and how well it fits into the entire context in which they are using it.”
- Alben [2]

“The user experience is the totality of end-users’ perceptions as they interact with a product or service. These perceptions include effectiveness (how good is the result?), efficiency (how fast or cheap is it?), emotional satisfaction (how good does it feel?), and the quality of the relationship with the entity that created the product or service (what expectations does it create for subsequent interactions?).” - Kuniavsky [35]

UXs are “based on instrumental (pragmatic) as well as non-instrumental (hedonic) system qualities” [22, 55]. Considering pragmatic system qualities such as usability is crucial [48] but UX goes beyond these aspects [55]. The terms usability and UX are often used interchangeably [15, 55]. But according to Hellweger and Wang [23] UX “should not be equaled to usability or user interface simply.” In the opinion of experts within the field usability alone can only achieve a limited level of UX [15]. We agree on this opinion and claim that usability is only one but essential part of UX. By understanding subjective and emotional experiences more meaningful and explicit targets for an application design can be facilitated [55].

Unfortunately most research in related fields is technology-oriented [1, 48] and literature related to HCI and UX has only a minor share [48, 55]. Väänänen-Vainio-Mattila et al. [55] and Queirós et al. [48] executed an extensive systematic literature review on UX research in UbiComp and Ambient Assisted Living (AAL).

Queirós et al. [48] analyzed 1048 articles in total. Only approximately 10%

¹²<http://www.allaboutux.org/ux-definitions>

(111) were related to direct user interaction and 29% were related to context-awareness within AAL. They found that “the [research] focus is still on the technology rather than on the person.” Many research projects concentrate on “how technology can be used in the AAL context instead of looking at the users’ needs and proposing ways to solve them” [48].

Väänänen-Vainio-Mattila et al. [55] collected initially 1016 article from various well-known databases. Only 75 (7%) papers were considered as relevant in the further course after applying filtering and selection criteria and content analysis. Field studies were the most often used approach for user studies among those papers. For evaluation purposes qualitative data gathering, more specific interviews, has been a prominent approach amongst the studies. Interesting findings of many studies about usage practices and design choice preferences enable insights into system use and needs for redesign purposes. Nevertheless, such findings provide no insights into actual subjective experiences. Many qualitative studies apparently did not attempt to dive deep into experience aspects. Thus, systematic, in-depth analysis of subjective and emotional experiences is scarce within these works. “Only 4 of the 75 papers were at a level of description of the subjective user experiences that foster deep understanding of how the [UbiComp] systems are experienced” [55]. According to them UX studies “need to take a broad spectrum of human experiences into account” to provide guidance for design and experiments for UbiComp and similar systems due to the novel and versatile technology involved [55]. The trend of demonstrating complex and multitude UbiComp systems became evident such as the Georgia Tech Aware Home [32]. Many studies included aspects to take the technology out from laboratory conditions into the real-world. Unfortunately “it is evident that the development of prototype devices and applications was conducted very much from the technical viewpoint” [55]. For this reason, the “central outcome of the field studies was very much to verify that the technology concept actually worked in the real world settings, rather than that it was valid for the actual end users and targeted contexts of use” [55]. The cause for the limited amount of research with focus on subjective, details aspects of UX amongst UbiComp studies is potentially caused by the strong technological and engineering background from which the UbiComp field originated.

UbiComp introduces paradigmatic changes such as implicit interactions, context-awareness, proactivity and engagement which may lead to new types of user experiences. Väänänen-Vainio-Mattila et al. [55] judge this potential of new experiences as worthwhile to be explored. Consequential they argue that a more fine-grained and thorough understanding of the types of experiences and how those are enabled by different technology features and design solutions is needed. For achieving this, qualitative, open-ended methods need to be applied in form of long-term studies with real target users within the actual context of use. In this way ecological validity and transferability of UX findings to similar system will be increased [55]. Unfortunately many UbiComp system in studies are still in rather immature stage. Technical challenges during implementation decrease time resources for user studies and lead to unavoidable usability problems during their execution. Thus, study outcomes are influenced negatively, even though the intended focus of the study was UX [55]. Queirós

et al. [48] furthermore points out that different technologies emerged from different research groups, lacking necessary interoperability. They therefore argue that efforts should be invested in the integration of existing technologies and interoperability.

The subsequent section introduces common methods for prototyping experiences for user studies and industrial design processes. Furthermore those methods will be elaborated towards their suitability for prototyping ILEs and similar systems.

2.4 Experience Prototyping

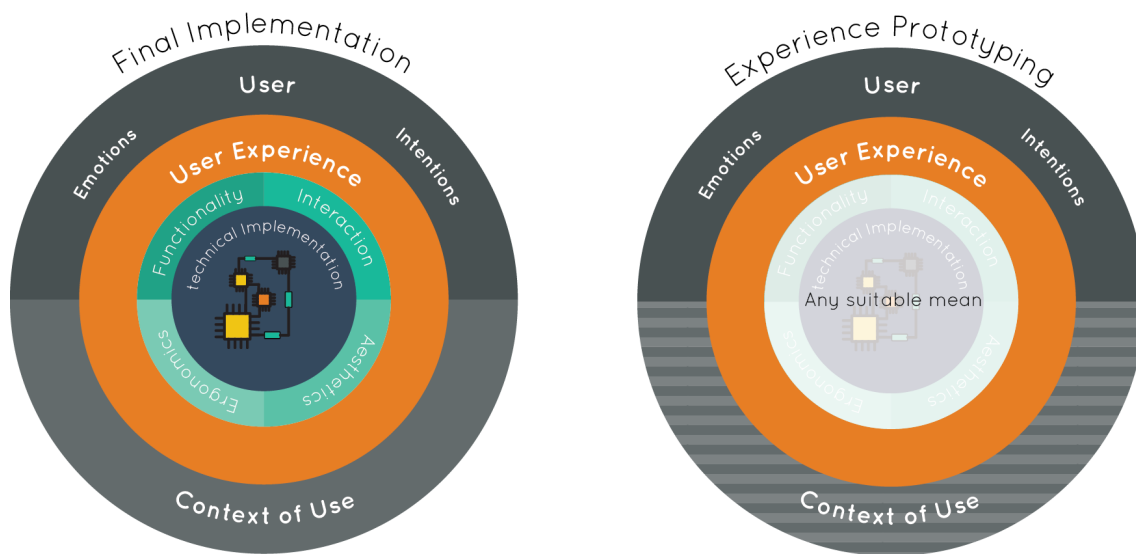


Figure 2.11: Illustration of Experience Prototyping

Prototyping is a key activity within the design of interactive systems and claims to be a key element of innovation in industry [10, 40, 42, 52]. Among others, Experience Prototyping (ExP) became a common practice since the term was first defined by Buchenau and Suri [10] from IDEO¹³ in 2000. According to them “[ExP] ... enables design team members, users and clients to gain first-hand appreciation of existing or future conditions through active engagement with prototypes.” They emphasize that an “Experience Prototype is any kind of representation, in any medium, that is designed to understand, explore or communicate what it might be like to engage with the product, space or system” in question (see figure 2.11). Prototypes in general allow “designers to demonstrate, evaluate or test an evolving design with minimal efforts” [54]. They can fulfill different functions [24], have different fidelity levels [33, 54], target different audiences and can be used as tool in participatory design [10]. Buchenau and Suri [10] considered preferably methods with active participation

¹³<https://www.ideo.com/>

to generate relevant subjective experiences instead of mediating a predefined one. They found that ExP contributes to a design process in three main ways:

1. Exploration and evaluation of ideas, generation of requirements and making design choices especially in an early design phase. Low-tech methods and basic materials are especially suitable.
2. Communicate ideas to different audiences such as fellow designers, technical developers, clients or end users. The fidelity level should be chosen appropriately to the design and development stage.
3. Help to foster an understanding about the essential factors of an existing application and its context. For achieving this more high fidelity prototypes are necessary.

As Buchenau and Suri [10] predicted, ExP established as a well-supported tradition within design practice over the last decade. The fidelity level of Experience Prototypes thus needs to raise to acquire more detailed and sophisticated insights if e.g. the deployment of a product is approaching [10, 55]. Unfortunately efforts with respect to time, work force and expenses are high for the creation of high fidelity prototypes [53, 54]. For this reason, low-fidelity ExP methods are way more common in the fast iterative design processes in industry. In the following some of these methods will be presented and exploited with regard to their applicability within a design process of an ILE.

[ExP.1] Collages & [ExP.2] Visualizations - *Collages* [50] are tools for the creation of semi-realistic mockups that help non-experts imagine unfamiliar devices already in an early design stage. Those documents should situate the physical objects in context. Trade-offs are often necessary for an effective early concept communication. On the one hand, the documents need to convey the full complexity of the concept. On the other hand, they should function as initiator for concept discussion.

Product/Service Visualizations [50] should be a single page mockup of key forms, relationships, and interactions. The visualization can be rough and imperfect. It can consist of a montage of photographs of existing sites with rendering of the concepts. In case that images are insufficient annotations can be added to provide additional information and remarks. The visualization should be specific and concrete so that it conveys problems, possibilities and relationships. In this way it should foster a discussion.

These two low-fidelity methods are used in an initial stage to discuss ideas and concepts and to discover potential problems. They are certainly also suitable for the design and development of ILEs in an early stage. Since ILEs consist of many components potentially multiple visualization are needed to foster discussion to an appropriate extent.

[ExP.3] Wireframing & [ExP.4] Paper Prototyping - A *wireframe* [53] defines the layout of a (part of a) Graphical User Interface (GUI) for e.g. a website, smartphone app or screen on another device. Hence, it functions to visualize and clarify structural aspects, terminology and navigation to a limited extent

due to its static nature. It can be created with pen and paper or on a white-board supported by special stencils or digitally. For digital creation standard graphics software such as Adobe Illustrator¹⁴, Sketch¹⁵ and OmniGraffle¹⁶, or specialized software such as Balsamiq¹⁷, Adobe XD¹⁸ and Mockflow¹⁹ can be used. It's creation should be fast and results can be rough and low in fidelity. *Paper Prototyping* [35, 53, 54] is the next evolutionary prototyping level. It is commonly used for the same application types as wireframing. In difference to wireframing it allows additionally the evaluation of dynamic interaction aspects such as navigation and workflow, page layout and functionalities [53]. In a user test developers or researchers "play the role of 'computer', manipulating the pieces of paper to simulate how the interface would behave" [53]. No explanations on how the interface is supposed to work is provided but only functionalities of the user interface are simulated [53]. In this way realistic interactions can be achieved [54]. All available materials such as paper, plastics or fabric can be used and crafted to a low-fidelity prototype [35]. It does not need straight lines or typed text, images or icons, color, and consistent sizing of components [53]. Therefore it is fine if it is rough or a bit messy [35, 53]. The integration of existing graphics, screen shots, photos or previously generated wireframes is nevertheless desirable [35, 53]. If the prototype incorporates content, then it "[needs] to [be] appropriate, detailed and meaningful to the application to preserve the fantasy" [35]. Snyder [53] elaborated on this: "If your user are accountants and you're testing a financial application, the number better make sense." As a rule-of-thumb the creation of the prototype should not take up more than one day [35]. Benefits of paper prototyping are that changes are easy and quickly realizable and no technological efforts are necessary. For this reason, "application designers, usability specialists and even the users themselves can create prototypes individually" [54]. However, a paper prototype does not deliver insights into technical feasibility, response times, layouts that require scrolling and design choices like color and fonts. Paper Prototypes and Wirframes are certainly not sufficient to precisely capture the interactivity of UbiComp environments [54] due to their visual nature. They are mainly used to prototype GUIs and other visual interfaces. However, Kuniavsky [35] points out that paper can also be used for scale models of scenery or objects. The two methods are certainly suitable for testing of ILE components such as the GUI a smartphone app.

[ExP.5] Physical Modeling ²⁰ as used in Industrial and Product Design is very important for the IoT, Smart Environments and ILEs. Since 'things' get connected functionalities and become avatars for digital data and representations, ergonomics and aesthetics of them contribute to the general UX of ILEs. First *mock-ups* in full-scale or scaled-dimensions can be used for explorations

¹⁴<http://www.adobe.com/products/illustrator.html>

¹⁵<https://www.sketchapp.com/>

¹⁶<https://www.omnigroup.com/omnigraffle>

¹⁷<https://balsamiq.com/>

¹⁸<http://www.adobe.com/products/experience-design.html>

¹⁹<https://www.mockflow.com/>

²⁰http://www.ruthtrumpold.id.au/destech/?page_id=57

and early user feedback. Non-functional *aesthetic models* carved from e.g. styrofoam, wood or clay or 3D-printed from plastics can be used in user testings for more precise and detailed feedback on aesthetics and ergonomics. *Scale-models* of environments and objects within are useful to illustrate proportions and relations.

It is evident that Physical Modeling on its own is not suitable for ExP. Nevertheless, we wanted to include it in this list based on the importance of the physicalness of objects in ILEs.

[ExP.6] GUI Prototyping is the final prototyping evolution from *Wireframing* and *Paper Prototyping*. Software like Axure, Framer or Adobe XD offer the possibility to create high-fidelity prototypes that are semi-functional and offer a polished look. GUI prototypes enable the evaluation of layout, structure, content and the graphical and Interaction Design (IxD) in detail.

They are similar to their predecessors certainly not suitable to provide the experience of an ILE. However, due to their high fidelity they can potentially be used as one component of a high-fidelity prototype of an ILE.

[ExP.7] Written Scenarios & [ExP.8] Storyboards - A *scenario* [35, 54] “tells a short story about people, situations, and how products introduced into that situation change people’s experience” [35]. The goal is by describing the details of the five aspects people, time, space, objects and circumstances/context to create a detailed story of a specific UX [35]. One concept or product can be subject of multiple scenarios. It is even advisable to create at least one scenario of everyday usage and another describing the situation of an edge case [35]. Scenarios enable designers to understand the everyday practices of their users [54], share understanding amongst them and mediate concepts experiences to clients and end consumers [35].

Storyboards [35, 50, 54] are an evolution of scenarios and complement each other. They visualize scenarios in sequences and transitions by graphic communication using images of people, objects, and environments but also diagrams, maps and symbols [50]. Equivalent to scenarios a concept “can have multiple storyboards describing possible experiences and interaction” [35]. They help imagine the various scales of a multi-scale interaction [35], its context and the cooperation of components [50]. Furthermore during the creation the designer needs to work through how the systems potentially fits together. The fidelity of graphics can additionally communicate the state of development of a component [50].

These two complimentary methods are especially viable in an early stage of concept generation [50, 54] also for ILEs. They also make it possible to draw visions of far futures and uncommon uses and interactions. Scenarios leave space for imagination of the user and therefore requires active involvement. Storyboards, on the one hand, limit this involvement by providing more restriction and, on the other hand, make the scenario more detailed and graspable by adding visual information. However, this limitations make it sometimes difficult to retrieve valuable user feedback [54].

[Exp.9] Enactment of Scenarios with designers, developers and end consumers and potentially props make it possible to better envision dynamic relations and context of a concept. Furthermore it involves the actors actively so that they can experience the concept more subjectively.

However, due to the complexity and continuous changing context [54] within ILEs a lot of influencing factors are left to the actors' imagination. Additionally the experience of an ILE can potentially only be enacted component-wise. This makes this method certainly suitable for early but not later stages in the development process of ILEs.

[Exp.10] Rapid Video Prototyping [35, 50] is not the creation of high-quality future visions as released on a regular basis by Microsoft and other IT companies. This method uses consumer-grade video hardware and software to create rapid video prototype using highly limited time, financial and workforce resources. These communicate and demonstrate concepts in action, and enable the quick exploration of ideas while ignoring technical details. Earlier prototyping results such as paper prototypes, scale and physical models, or scenarios in form of enactment can be incorporated. The video can also be realized by stop-motion animation with e.g. LEGO.

Rapid video prototyping has the advantage over live enactment that the experience can be enhanced by post editing and it can be used for mediating the idea to a bigger audience. However, since the audience is not active involved and the experience is predefined their feedback is potentially not as valuable since they did not have a subjective experience. Therefore it only seems suitable for UX design of ILEs to a limited extent or in early stages.

[Exp.11] Wizard of Oz [50, 54][35, p. 228] is in general a test setup in which a person (the *Wizard*) observes the input of a system (e.g., user's actions) and simulates the system's responses in real time" [54]. This enables the simulation of complex and native interactions such as speech input or gesture control or the interaction with an intelligent system.

It is necessary to implement and test a second interface for the wizard. The effectiveness of the prototype depends on the wizard's level of task understanding and skill to control the interface [54]. Unfortunately "the complex and multimodality interactions make the high-fidelity implementations [of ILEs] very difficult and also make in situ testing challenging" since human perception is limited. But the Wizard of Oz technique can be used to simulate complex components in a high-fidelity Experience Prototype of an ILE that would require high efforts to be implemented.

[Exp.12] Functional Component Prototyping is used to make the experience of specific functionalities or functional components available. It is basically a form of Technical Prototyping, as described in chapter 2.1, using easy-to-use prototyping platforms such as Arduino. The tools can be used for the creation of complete high-fidelity prototypes of a product but due to the high efforts necessary in most cases only components are implemented. Prototyping a whole ILE would require even higher efforts that are most likely not affordable and realizable in a fast iterative design process. In comparison to a functional

prototype of a stand-alone product in an ILE furthermore the technical and semantic communication of components needs to be implemented. Incorporating, even only a rule-based, intelligence increases the required efforts even more. Therefore, the creation of a complete functional prototype of an ILE for UX design and research purposes seems unadvisable with conventional tools for technical prototyping.

The previous elaboration on common ExP methods in industry and research as shown that they are not suitable for providing the UX of an ILE in its whole. Low-fidelity methods like collages, visualization, scenarios and its various derivatives are suitable for ideation and concept generation in an early stage. Other methods like wireframing, paper and GUI prototypes and physical modeling can be used to design and develop the individual UX of components within an ILE. High-fidelity component prototypes in form of sophisticated physical models, GUIs or functional components can potentially be used for the creation of a high-fidelity Experience Prototype of an ILE.

As Buchenau and Suri [10] already indicated it is challenging “to provide an early, low-fidelity improvisation prototype of sufficiently robust nature that they can have an experience in a naturalistic context without supervision.” The heterogenous and dynamic context of ILEs make it difficult to generate and predict scenarios and problems. The prediction of users’ behaviour within this context is difficult as well [54] since UX is highly subjective [10]. Therefore it is necessary as Väänänen-Vainio-Mattila et al. [55] proposes that user studies are “conducted with real target users, in the real contexts of use and in long-term use” to increase the validity of UX findings. Current prototyping methods and tools are unfortunately not supporting the implementation of prototypes that are suitable for such user studies. Even though researchers have been working on rapid prototyping tools for systems similar to ILEs they are still in an early stage [54].

2.5 Problem Statement

This section describes in more detail the problems faced within this work and their origin.

As shown previously current products and services are not meeting the desire of end consumers (see chapter 2.2). They would prefer more intelligent cooperative systems that are more easy-to-use over current products [25]. This is caused on the one hand by missing efforts of the industry on semantic interoperability (see chapter 2.1). On the other hand UX design for ILEs is lagging behind the UX knowledge on stand-alone applications from our point of view. We see this confirmed by the findings of Väänänen-Vainio-Mattila et al. [55] (see chapter 2.3). Their literature overview illustrates that in-depth UX research for UbiComp systems is scarce. The origin of the field from a technical and engineering background is one reason. Another reason is that technical challenges during implementation of systems for user studies and their execution decrease the time span to gather and the quality of results [53, 55]. They find that long-term user studies in real context with real users are necessary to increase the validity of UX findings. Therefore the creation of suitable and stable Experience Prototypes is needed. We evaluated common prototyping methods in UX design towards their suitability (see chapter 2.4). Due to the heterogeneity, complexity and dynamics within ILEs, their context of use and new ways of interaction [54] none of the common prototyping methods are suitable to capture the experience of an ILE in its whole with reasonable efforts. Additionally Buchenau and Suri [10] claim that “solutions, and probably even imagination, are inspired and limited by the prototyping tools we have at our disposal.”

For this reason, this work attempts to ...

[MainObj] Facilitate time- and cost-efficient functional Experience Prototyping (functional ExP) of ILEs for UX research, design and development.

Following questions need to be clarified to approach this problem statement:

- [Q.1] - What is the desired UX of ILEs?
 - [Q.1.1] - What are the dimensions of UX?
 - [Q.1.2] - To which extent can an UX outline for ILEs be defined?
- [Q.2] - What is a typical design and development process in UX research and industry?
 - [Q.2.1] - Does this process need to be adapted when incorporating functional ExP?
- [Q.3] - What are the specifications for a tool that supports a design and development process that incorporates functional ExP?
 - [Q.3.1] - How to map the identified specifications to functional tool components?

2.6 Functional Experience Prototyping

We understand functional ExP as following:

Functional Experience Prototyping is the prototypical implementation of a User Experience (UX) that is equivalent to the UX of the intended final product. The prototype will most likely not contribute to the technical realization of the final product but is not restricted in this aspect.

Functional Experience Prototyping enables thus the fast and easy creation of high-fidelity Experience Prototypes of a concept. The resulting prototype provides similar UX and functionality in comparison with the intended final implementation. Nevertheless, the technical details, efficiency, scalability, etc. of the implementation are beside the point. Except if the UX of the prototype would differ from the intended final implementation in such a way that findings of user studies carried out using the prototype would become invalid. Therefore, the technical implementation of the prototype will most likely not contribute to the final deployed implementation. However, we don't want to limit the term functional Experience Prototyping (functional ExP) here, since innovations in development tools are steady and could lead to a fusion of functional ExP and Technical Prototyping to some extent.

Figure 2.12 illustrates the differences between functional ExP and other types of prototyping and an final implementation of an application.

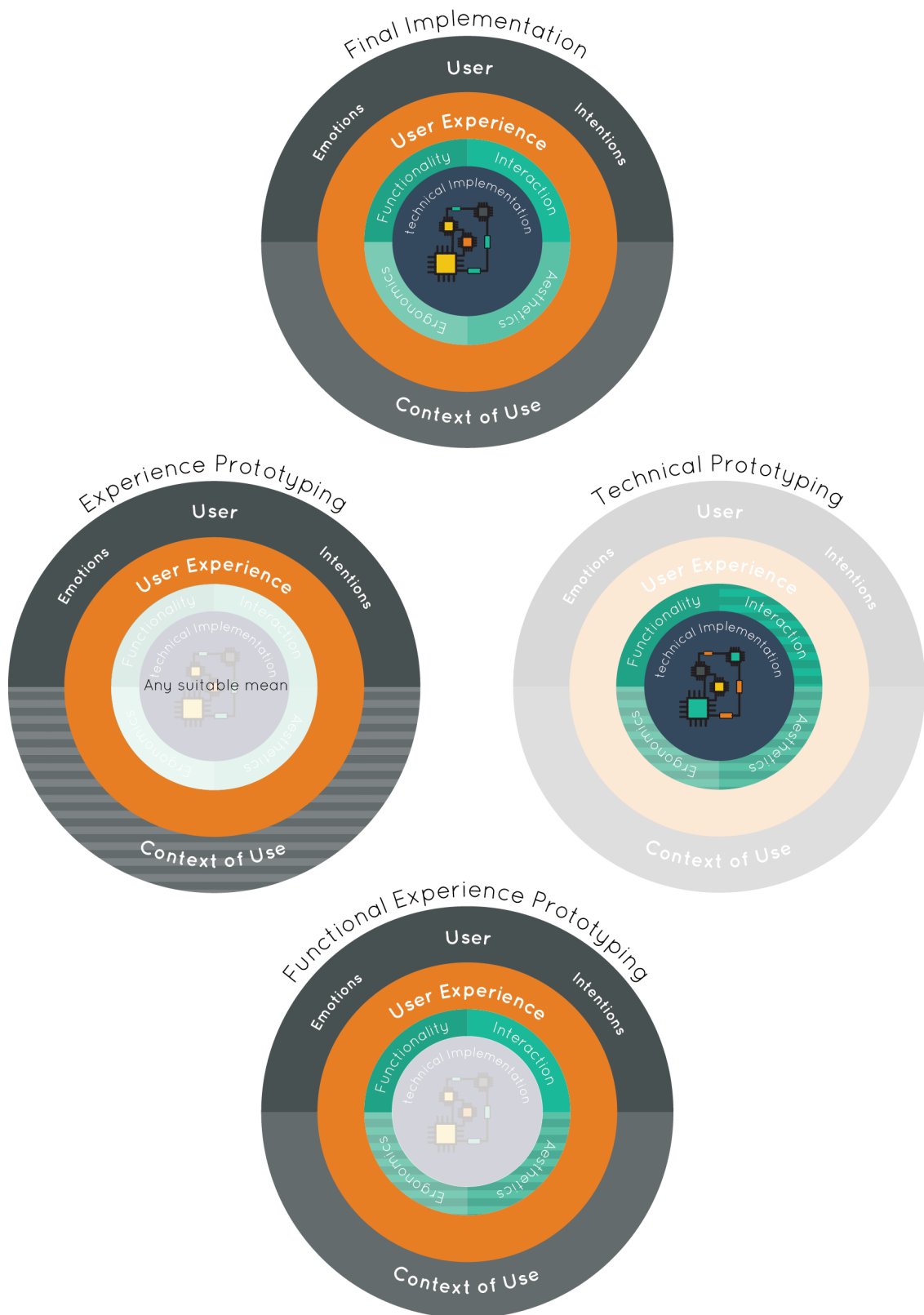


Figure 2.12: Overview of different types of prototyping activities.

Chapter 3

Desired UX

The desired UX of ILEs needs to be known so that it can be realized within a functional Experience Prototype. For this reason, this section aims to define an outline for desired UXs of ILEs.

First it needs to be clarified which dimensions are influencing UXs and therefore need to be included in a generic UX outline. Afterwards, the resulting UX outline of ILEs is presented by elaborating on the selected dimensions. However, before diving into UX dimensions and outlines two important aspects of designing UXs need to be mentioned. Next to the external factors it is important to define the type of UX that the designer is aiming for and the scope of the UX considered within the design. The next paragraphs will give more details about these two aspects.

Types of UX - The field of psychology defines three different types of motivation: utilitarian, hedonic and eudaimonic [47, 51]. We borrow these terms to define three different non-exclusive types of UX:

- **Utilitarian** experiences aim for the pure fulfillment of the user need or Intention of Use without considering other dimensions such as the emotional state of the user or the Context of Use. This would correspond to a pure functional application that uses the most common and easy-to-implement way of interaction with the user e.g. keyboard and mouse or nowadays touch screens.
- **Hedonic** experiences aim for fun and enjoyment to improve well-being. Next to the fulfillment of the user need other aspects such as usability as well as aesthetics are considered in the design process. Making the experience not only intuitive by for example using body gestures but also enjoyable by e.g. funny ways of feedback.
- **Eudaimonic** experiences try to add to or support the meaningfulness of a system. Eudaimonic well-being concerns aspects such as self-acceptance, personal growth, purpose of life and more. An experience can support this by considering e.g. the emotional state and emotions of a user during the use of a system.

Important to mention is that these types of UX need to be differentiated from the Intention of Use of a system. A system that aims on fulfilling a hedonic need such as a game can implement a pragmatic/utilitarian experience. However,



Figure 3.1: The different scopes of UX design.

aiming additional for a hedonic experience would most likely improve the fulfillment of the hedonic user need. In our opinion a hedonic or eudaimonic experience is always based on the fulfillment of the user need thus a utilitarian experience.

Scope of UX Design - It is important, especially with the background of ILEs, to define the scope of the UX that is intended to be designed in the very beginning. The following list provides more detail about different experience scopes.

- **Brand Experience** [38] includes the interaction with a whole company, all its products and services. A positive or negative brand image can influence the user experience. A image of a high valued brand can outweigh minor design flaws. A negative image can even prevent the whole experience to happen at all. In the same way as an experience with every service or product influences the brand experience. The brand experience forms the initial expectations towards a product.
- **Environment Experience** is the experience of the interplay of different products and services potentially from various manufacturers. They pursue shared objectives such as the satisfaction of user needs. All components have their own individual user experience. However, it needs

to be insured that they form one monolithic Common Operating Picture (COP) [20].

- **Service Experience** [38] is the experience of one service that involves multiple components. All components contribute to the achievement of one objective. The experience within a service is distributed onto all components that come into contact with the user. An environment experiences could be assembled out of multiple service experiences. However, the interplay within a environment experience is potentially way more dynamic.
- **Product Experience** [38] is the experience with one specific artefact. A product experience can be part of service, brand, or environment experiences.

Figure 3.1 illustrates the relations between those different UX design scopes.

3.1 User Experience Dimensions

The term User Experience (UX) has been already introduce in section 2.3. This section elaborates on the dimensions that influence and built-up a UX from a more general perspective. The identified dimensions will be used to form an outline for desired UXs of ILEs in the further course of this chapter.

An “experience is a very dynamic, complex and subjective phenomenon” [10]. Experiences, even of simple artifacts, live in a complex dynamic context of people, places and objects [10]. According to Law et al. [38] experts in the field agree that “contextual factors are important influencers of UX.” Hassenzahl and Tractinsky [22] define three dimensions of UX:

1. user’s internal state (predispositions, expectations, needs, motivation, mood, etc.)
2. characteristics of the designed system (e.g. complexity, purpose, usability, functionality, etc.)
3. context within the interaction occurs (e.g. organizational/social setting, meaningfulness of the activity, voluntariness of use, etc.)

Hellweger and Wang [23] propose a conceptual UX framework based on this three dimensions that include following aspects: Product properties, context, usability, cognition, purpose, (user) need. Derived from the definitions of predecessors [22, 23, 38] we defined our own UX dimensions as shown in figure 3.2. In the following a short elaboration on the various aspects is given:

- **User’s Internal State**
 - **Expectations, mood, emotions, meaningfulness of activity, prior knowledge** are important factors that influence the user experience. However, those are highly subjective and depend on the context that the user is currently in.

- **Intentions of Use** summarize the user needs and motivations to use a specific application. The objective or user need that the intended application should fulfill is the most fundamental aspect of an UX [18].
- **Context of Use** includes temporal, social, physical, technological and task-specific aspects of the context that the system is used in.
- **System Characteristics** such as functionality, usability and complexity influence the perceived user experience of the system. The purpose of the system should meet the Intention of Use if a positive user experience is desired.

In the subsequent section the gathered insights on UX from this section will be used to draw a generic outline for UXs of ILEs.

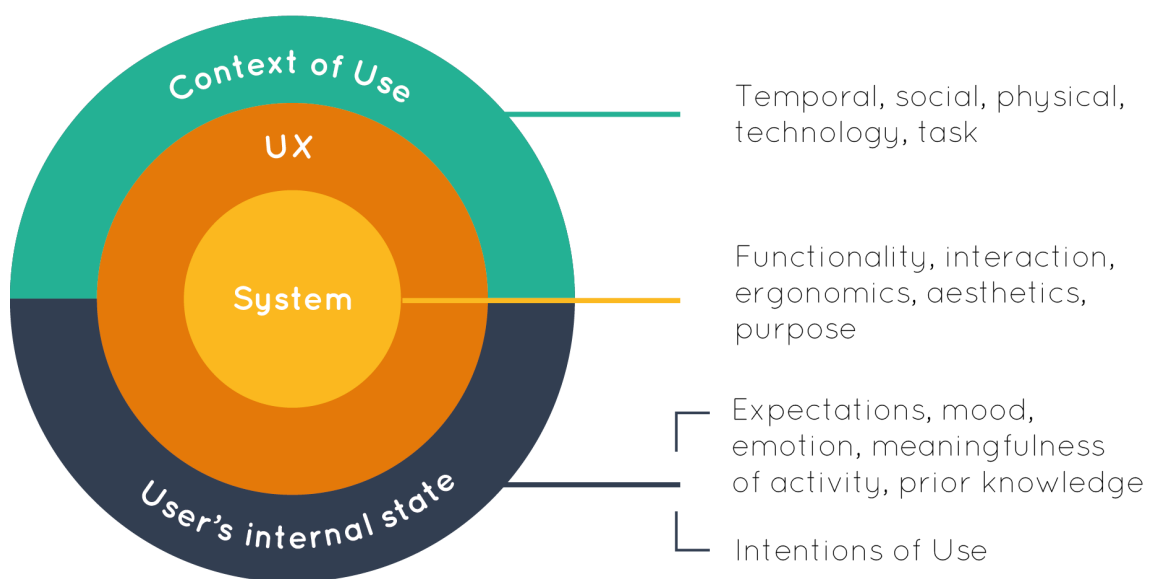


Figure 3.2: The three dimensions of UX.

3.2 UX outline for ILEs

This section defines an outline for UXs in ILEs by elaborating on appropriate dimensions of UX. The previous section introduced these dimensions. Due to the subjective nature of some of the aspects concerning the user dimension those can not be generalized and therefore not considered in this outline. However, the subsequent section provide details concerning UXs of ILEs with respect to the Intention and Context of Use, specific aspects of UX and IxD that originate from system characteristics as well as typical components that can be assembled to ILEs.

Before diving into the different dimension of the UX outline the scope of the UX in question should be clarified. As introduced in the beginning of this

chapter the UX of an ILE is an environment experience. Such an experience is not just the sum of the individual UXs of the separate components. The environment experience incorporates additionally factors such the interplay of explicit and implicit interaction with assembled products and services. Figure 3.1 illustrates this constellation.

3.2.1 User's Internal State

It is evident that many aspects of the internal state of a user are highly subjective and vary in between each individual. Aspects such as user's expectations, emotions, moods, meaningfulness of an interaction and prior knowledge are therefore highly dynamic factors as an result of interplay between the user's personality and the current Context of Use. Therefore, user evaluations, as this work attempts to enable, are necessary to consider theses aspects within a design and development process of an application. Nevertheless, one part of the user's internal state can be more generalized for ILEs as shown in the following.

The need that the user intends to satisfy by using an ILE application is the foundation of every UX. An unfulfilled user need has potentially such a negative influence on other UX dimensions, in our opinion, so that the intended designed UX is not experienceable. Subsequently, we attempt to summarize the *Intentions of Use* for ILEs. The overall goal of ILEs is an increase in life quality of its user in terms of comfort, health, environmental impact and more. Considering the collection of application example presented in section 2.2 and customer surveys [25] we derived more specific *Intentions of Use* as listed below. Figure 3.3 provides an overview of these as well.

- **I.1 - Efficiency** of resources and activities of the user. ILEs can facilitate a more efficient use of resources such as energy (e.g. smart thermostats [EX.11]), water, food or shared goods (e.g. car-sharing). This leads to higher cost efficiency as well. They can furthermore improve efficiency of time efforts and logistics within user activities to achieve e.g. a better work-life balance.
- **I.2 - Empowerment** - ILEs can empower user to e.g. live on their own longer than without the system. They can improve safety by monitoring user activities, take care of chores that user can not execute anymore themselves caused by e.g. their age, and connect users to remote living relatives in every day life ([EX.31]). This concept is also known as AAL. Other forms of empowerment are for example improved educational possibilities by e.g. interactive toys (e.g. Cognitoys [EX.36]).
- **I.3 - Comfort** - Taking over chores in every day life can also increase comfort of the users. This increase can also be achieved in new ways of intelligent activity support such as lighting systems (e.g. [EX.10]) that automatically adapt to the user activities such as reading a book. ILEs can furthermore enhance and support the experience of entertainment.

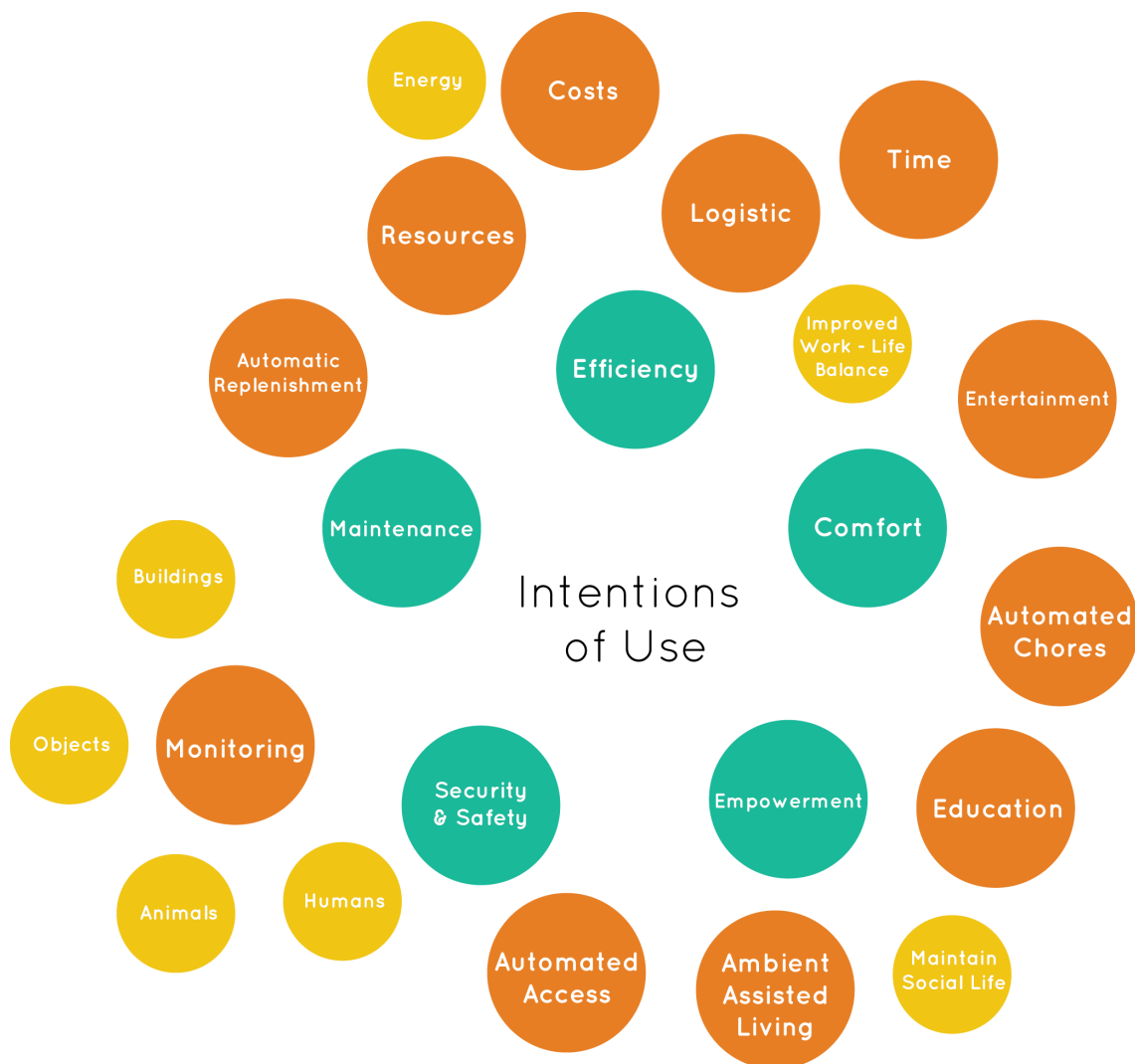


Figure 3.3: General Intentions of Use for ILEs.

- **I.4 - Security & Safety** of users and their property can be enhanced by the advanced surveillance and monitoring capabilities of ILEs. For example user can be recognized and by this access can be granted without the need of a physical object like a key (Smart Locks [EX.20]). Continuous monitoring of user and infrastructures such as building can furthermore increase safety e.g. against accidents.
- **I.5 - Maintenance** - The continuous monitoring of infrastructures and devices can also simplify and improve maintenance. In comparison to fixed maintenance schedules systems to an ILE can indicate when maintenance is really necessary. Smart devices can additionally also order replacement parts themselves (e.g. [EX.18.2]))

In our opinion it is important to mention that user needs as well as purposes of ILE applications can intersect with multiple of the generic Intentions of Use introduced above. For example increased safety by continuous monitoring also

empowers user to live longer on their own. Intentions are of course highly dependent on the user e.g.: Some users may perceive daily cooking as a chore and would prefer if an ILE application would take over this task. Other users may perceive this task as delightful and would like to take care of it as long as and whenever possible themselves. This should be considered when designing and developing an ILE application.

3.2.2 Context of Use

As elaborated previously the *Context of Use* is a major factor for the user experience of ILEs. One general concept that needs to be considered in ILEs is that the user constantly carries his micro environment with him in form of e.g. personal devices. In optimal case this micro environment should integrate seamlessly with the surrounding macro environment. Furthermore it is important to consider all contextual dimensions as identified by predecessors [38]:

- **CO.1 - Temporal** context can influence the way a system reacts to a user. Supporting the user after getting out of bed in the morning is most likely desired by the user. However, same supporting actions can be highly inappropriate when getting out of bed in the middle of the night or after nap in the afternoon.
- **CO.2 - Social** context creates challenges for the behavior of a system. The behavior of a system like setting the preferred air temperature within a room should consider all persons present. It is still a matter how to solve such conflicts. Additionally, some behavior of a system that is desirable, when the user is on his own, is potentially not desirable when other people are around.
- **CO.3 - Physical** context is a mayor factor that needs to be considered within ILEs. ILE applications should consider the physical context of its users as well as devices. This enables so called implicit interactions with an application. This term will be discussed in more detail in section [3.2.3.2](#).
- **CO.4 - Technology** infrastructure surrounding the user and other devices should be incorporated if worthwhile. For example the playback of a video should happen on the most appropriate screen available in the surrounding of the user instead of e.g. a smartphone display. Information from additional devices could be requested to increase confidence of e.g. the recognition of user identities or activities. The consideration of the technological context is the foundation that enables the integration of the micro and macro environment of the user.
- **CO.5 - Task** context is important as well. An subtask performed by the user can have different meaning and impact for an ILE depending in the main task executed. Boiling water with an electric kettle after getting out a cup implies another main task than getting out a pot and set it on the

stove previously. By recognizing that the user intends to prepare some tea the electric kettle could adjust the final water temperature according to the chosen sort of tea. In this way an improved result such as better taste could be achieved during the infusion of the tea.

Same as for the *Intentions of Use* the *Context of Use* is always a mix of all aspects. All factors are interrelated and can hardly be seen in an isolated manner.

3.2.3 System Characteristics

The third UX dimension that was derived and set within this work are the characteristics of the system in question. These are certainly dependent on the actual application and can hardly be generalized. However, we identified general characteristics of IoT systems within a preparatory study. These characteristics were previously presented in the technical background section 2.1. In the further course of this subsection first the impact of the system characteristics on the UX and IxD of ILEs will be discussed. Afterwards, the composition of ILEs will be analyzed with a set of abstract components for ILEs as a result.

3.2.3.1 User Experience Aspects

In course of this study it was already shown that the UX of stand-alone application (product/service experience) differs from the UX of ILEs (environment experience). In the following important UX aspects for ILEs are listed. These were partially derived from the system characteristics defined in chapter 2.1 and gathered from work of predecessors.

- [UX.1] - **UX is distributed and dynamic** - ILEs have dynamic N-to-N relations (→ [CR.3]). The interaction with one ILE application can happen over multiple interfaces by one or multiple users, the functionality can be distributed and new components can dynamically be added to the system. Furthermore, the *Context of Use* can change dynamically → [CR.4]. [50, p. 7] [21, p. 112]
- [UX.2] - **UX depends more on the Context of Use** in comparison to UX of stand-alone products. The *Context of Use* within ILEs is highly dynamic and diverse (→ [CR.4]). [50, p. 11] [35, p. 25] [54]
- [UX.3] - **UX can be discontinued** - Since ILEs are asynchronous, not all devices are always online due to e.g. energy management → [CR.5] [TCL.4]. Additionally, considering [UX.1], user can stop the interaction flow on one device and continue on another that offers the same functionalities → [CR.3]. [50, p. 8] [21, p. 112]
- [UX.4] - **Responses & feedback can be delayed or lost** - ILEs can experience latencies and loss of messages since the networks and the internet are not 100% reliable → [CR.6].

- [UX.5] - **UX can be unexpected and surprising** - Taking [UX.1] into account it is evident that data can be manipulated and actions can be triggered over multiple interfaces either by another human, by rules or by intelligent agents. Rules can potentially have been set by another person or the user himself. The user may also forget about rules that he had set himself in the past.
- [UX.6] - **UX can be shared amongst people** - An ILE is most likely a shared space in which multiple people live or at least are sometimes present. Therefore major parts of the UX of an ILE will be shared amongst those.

3.2.3.2 Interaction Design Aspects

The system characteristics and their influence on the UX have also an impact on how interaction should be designed for ILEs. Two different kinds of interactions are present within ILEs: Explicit and implicit interactions [54]. Subsequently, first a list of aspects that are important to consider when designing interaction within ILEs in general is presented. Afterwards, the terms explicit and implicit interaction will be explained in more detail and specific design aspects for those types of interaction will be pointed out.

Interaction within ILEs ...

- [IxD.1] - ... **should (at least partially) be available when disconnected.** So that even if devices or services are shortly or completely disconnected part of the interaction still remains available → [UX.3].
- [IxD.2] - ... **should be robust against discontinuity in interaction.** An ILE should be able to cope with situation where interaction is interrupted and potentially picked up on a later point in time → [UX.1] [UX.4] [CO.1].
- [IxD.3] - ... **should be robust against other associated devices being disconnected.** Not all devices within an ILE are always online due to e.g. energy management → [UX.3]. In such cases interaction flow that require those devices need to be robust against this fact.
- [IxD.4] - ... **should be robust against conflicts between user interests.** ILE are spaces that are most likely shared amongst multiple people. Therefore the interest of one user (e.g. preferred room temperature) can be in conflict with the interest of other users → [UX.6]. An ILE needs strategies how to cope with such conflicts preferably autonomously. Depending on the type of interest averaging, merging or assigning priorities could resolve the conflict. Asking the users to resolve the conflict themselves could also be an option in a social context where the users know each other.
- [IxD.5] - ... **should be always transparent for the user.** The state of an interaction should always be clear to the user e.g. by incorporating

more feedback as in “sending request”, “request arrived”, “request executed” [50] → [UX.1]. Additionally the user should have insights about all implicit interaction happening by e.g. providing an overview about those.

Explicit Interaction are executed by the user with direct intention to manipulate the system. Explicit interaction are well-known to user nowadays in form of interactions with e.g. keyboard and mouse, touch screens or speech interfaces. In the following some aspects that were derived from general system characteristics and UX aspects stated earlier are listed.
Explicit interaction within ILEs ...

- [IXD.Ex.1] - ... **should consider the Context of Use.** The interaction should be adapted to the context of use → [UX.2]. Controlling an ILE via speech at home seems desirable to a user. But in public many people dislike using speech input, same is true for synthesized speech output. A graphical user interface (GUI) may in that context be more suitable. Continuing this thought, the GUI should automatically adapt between in home and outside context, offering primarily the more important functionalities for the Context of Use.
- [IXD.Ex.2] - ... **should be meaningful within the Context of Use.** The interaction type should be chosen considering the context. For example if the user is in a room and wants to lower the window blinds. Controlling them by point and speech control (→ “lower”) is a way more meaningful interaction in this context than using a smart phone app to do the same. Furthermore, devices that evolved from traditional devices (e.g. smart thermostats) should offer similar functionalities and interaction possibilities as their traditional equivalents. For example some smart thermostats on the market lack direct interaction possibilities and need to be mainly controlled via a smart phone app.
- [IXD.Ex.3] - ... **should be as modular as possible.** It is a sensitive matter how to divide and distribute the interaction considering [IXD.Ex.1] & [IXD.Ex.2]. This needs to be done in such a way that one module still offers a suitable amount of functionality by itself → [IXD.1] [UX.1].
- [IXD.Ex.4] - **should not be (physically) exhausting for the user.** Interaction types that require physical movement or holding of poses should be limited and not stress or exhaust the user except stimulating physical exercise is intended.

Implicit Interactions are executed by the user unconsciously through his every day behavior such as change in location or activity and are interpreted by a system to react in an appropriate way. Implicit interactions are used within context-aware applications or systems such as ILEs but also stand-alone applications. Context-aware applications are still a minority but steadily gain more interest.

Implicit interaction within ILEs ...

- [IxD.Im.1] - ... **should consider the entire Context of Use.** Implicit interaction is by its nature based on the context surrounding the user and the system → [UX.2]. Nevertheless, all contextual dimension as defined in section 3.2.2 need to be considered to interpret the Context of Use appropriately.
- [IxD.Im.2] - ... **should provide a feeling of being in control to the user.** The user should have control over the intelligence, inputs, outputs and his data. Even if it is not the case that the user is in complete control, the system should still provide the feeling that he is. Microsoft attempts to provide control over his data used by their Intelligent Personal Assistant (IPA) Cortana with Cortana's Notebook¹: A written transcript of all information collected on the user. Providing control over results of implicit interactions could be implemented by granting veto rights for those to the user → [UX.1].
- [IxD.Im.3] - ... **should not disturb or distract.** The user's every day behavior should not be influenced unintentionally. Excluded are intended actions e.g. reminders or behavior adapting applications.
- [IxD.Im.4] - ... **should not take the user too much by surprise** Outcomes initiated by implicit interaction can be unexpected for the user caused by the complexity of and intelligence within an ILE → [UX.1]. The system therefore should be transparent (→ [IxD.5]) and autonomous actions with major impacts should be announced or approved by the user.

¹<http://windows.microsoft.com/en-us/windows-10/getstarted-cortanas-notebook-mobile>

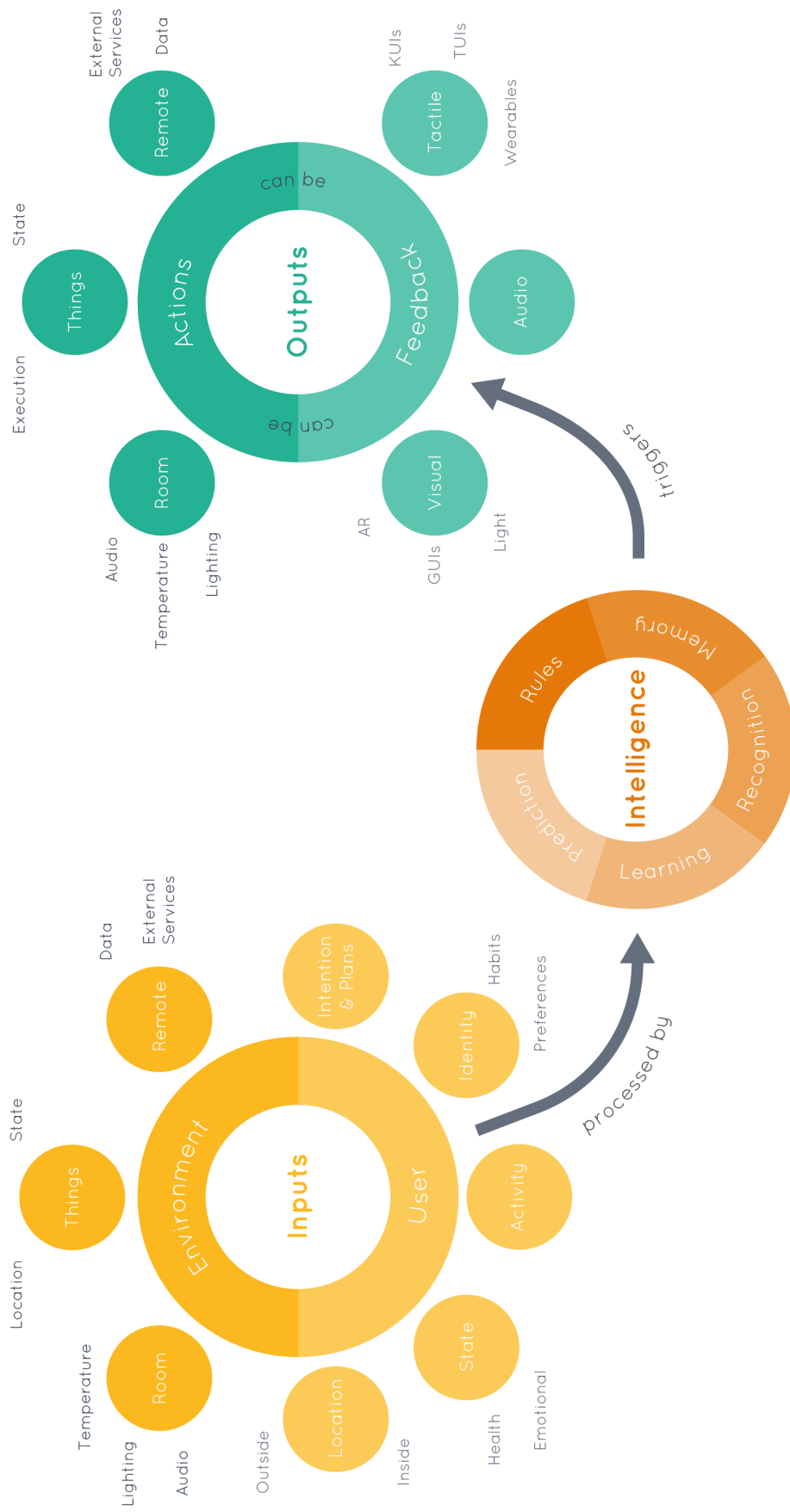


Figure 3.4: Overview of possible components of an ILE categorized into inputs, intelligence and outputs.

3.2.3.3 Components of ILEs

As last part of the UX outline and the system characteristic dimension we define a set of abstract components from which an ILE can be composed of. The components chosen for an ILE form the base of the UX and are therefore essential for our UX outline. Figure 3.4 shows an overview of the set of collected components. This overview concentrates mainly on components involved with **implicit interactions**, since this type of interaction sets ILEs apart from most current applications. The three main component categories are **inputs**, **intelligence** and **outputs**.

On the one hand **inputs** can originate from the environment. That can be the current room or space the user is in, connected things within this environment or remote services and devices. Inputs are on the other hand related to the user: his identity, intentions and plans, activity, emotional and health state, and location. **Intelligence** observes and logs all those inputs. Aspects such as the intention, activity and emotion of the user need to be recognized by an ILE. Based on all this collected and categorized data an ILE should learn how to react in the future and what the habits and preferences of the user are. An ILE can derive rules itself based on this learned insights but also a user can enter rules for the system himself manually. Those rules will then in the future trigger **outputs** in form of appropriate actions and feedback for the user. Feedback can be either visual implemented by e.g. simple light changes or GUIs, tactile implemented by e.g. Tangible User Interface (TUI)s, Kinetic User Interface (KUI)s or wearable devices, or audio. Of course different compositions incorporating all three feedback channels are also possible. The execution of actions can also represent a kind of feedback, if executed in the presence of the user. Actions can be performed by the environment/room itself, things within this environment or by remote devices or services.

As mentioned previously this overview concentrates more on implicit interaction. Nevertheless, **explicit interactions** are existing within ILEs as well. Very common methods here are GUIs on touch-enabled devices and speech interaction with IPAs as implemented in Amazon Echo and Alexa (see [EX.27] figure 2.9 p.16).

3.3 ILE Use Case Examples

In this section six use case examples of ILE applications developed during the course of this work are contained. They draw new application scenarios that have not been observed by us, at least in the extent defined here. On the one hand these use case examples provide insights on how we imagine future ILE applications and their UX. On the other hand they function as another input for the specifications of a tool that supports functional ExP defined in the further course of this work.

The Extended Memory - ``Where did I leave ...?' [UC.1]

Functionality - The ILE gives hints to allocate lost objects within the home of a user.

Description - The user can interface the service via an explicit interaction with e.g. his IPA by asking „Where did I leave my glasses?“, „... book?“, or „... key?“. The feedback can be given in various different ways. The IPA could reply with „You left your glasses in the living room (somewhere near the TV)“. Additionally or alternatively the object itself could light up, ring or vibrate.



Required Technology - An implementation of this service would require a location tracking service most likely based on BLE to realize the basic functionality. For the input a voice interface or a GUI are potential options. Feedback can be generated by the same voice interface, as part of an IPA, or by specialized connected hardware attached to the object in question.

Figure 3.5: “Where did I leave ... my glasses?”

Source: [Search For Happiness](#)

Context-aware Reminders [UC.2]

Functionality - The ILE reminds the user in a unobtrusive ambient manner about various issues important to the user based on context-awareness.

Description - The ILE tracks the user activity and his context. Additionally it has access to his calendar and other data feeds. Based on these inputs the ILE learns about the daily routine of the user and his activities. This enables the ILE to remind the user to perform actions critical to his safety and health such as turning of the oven after cooking, brushing his teeth and taking medication. Less critical issues such as watering the plants, call back relatives, or to not to forget an umbrella can

be a treated matters as well. Furthermore more conventional reminders on appointments or scheduled tasks can be incorporated. Such an application is especially interesting for elderly people in the context of AAL.



Figure 3.6: Remembrall in the Harry Potter Movie.

Source: [Warner Bros. Entertainment Inc.](#)

Important for this target group is that the application does not take a paternal role. Patronizing behavior is often experienced as unpleasant and is not endorsed by elderly people. Reminders could be mediated in rough categories like different colors within the environment such as the “Remembrall” (see figure 3.6). However, the reminder should be displayed on the most suitable medium closest to the user. Additional information on the reminder could be provided via an companion app or the IPA.

Required Technology - For the realization of the basic functionalities activity recognition, sensor infrastructure, internal and external data feeds and machine learning are required. Explicit inputs are not present in this example. Reminders are triggered indirectly by the user and his activities and sensor and data inputs processed by Artificial Intelligence (AI). More rudimental triggers are scheduled appointments and tasks. Output can be connected ambient screens in form of Smart Home hardware, specialized hardware or displays in form of interactive artworks. For displaying the reminder in the right place user identification and localization are necessary as well.

Context-aware Lighting System [UC.3]



Figure 3.7: Ario Smart Lighting system over a day.

Source: [Ario](#)

Functionality - The ILE automatically adapts the lighting conditions inside to be appropriate with respect to time of the day, lighting conditions outside, and the activity and schedule of the user similar to the Ario Smart Lighting System (see figure 3.7)

Description - The ILE automatically turns on the light when the evening approaches or the lighting conditions outside are dark. It also adapts the lighting to the activities of the user such as dimming the light when

watching a movie or optimizing the light conditions for reading. It could also take over functionalities as offered products such as the Philips Wake Up Light by simulating a sunrise to ease getting out of bed. Since the ILE could be connected to a sleep sensor such as SleepSense (see [EX.15] figure 2.9 p.16)

this could even happen in the most appropriate sleep phase of the user. The opposite functionality to dim down the light to prepare the body in the evening to go to sleep could be implemented as well.

Required Technology - An implementation of this system would require activity recognition, sensor infrastructure, external data feeds, machine learning capabilities and a connected lighting infrastructure. This system would in every day use mainly be based on implicit interaction. However, explicit interaction possibilities should be provided as well so that the system can learn the user preferences. They are additionally useful if the system is mistaken or the user has exceptional demands. In this way the user is furthermore always in control of the system.

Context-aware Interfaces [UC.4]

Functionality - Interfaces that simplify interaction by adapting to the Context of Use.

Description - An interface e.g. a GUI adapts its appearance and offered functionality to the Context of Use. Incorporated in this context is time, location, environment, the user and his activity and identity as well as social and technological factors. For example a coffee machine GUI could offer directly the preferred choice of a user depending on the day time instead and his preferences and habits instead of requiring him to enter type, strength, added sugar and milk himself. Another example is a Smart Home app that offers different functionalities when the user is at home in comparison to on the go.

Required Technology - For realizing such an interface user identification, localization, activity recognition and external data feeds are necessary. Furthermore machine learning is required to learn about preferences and habits of the user. Depending on the kind of interface additional technical components are needed.

Context-aware Objects [UC.5]

Functionality - Objects that change their behavior and/or functionality depending on the Context of Use.

Description - An object like a speaker for the playback of music could automatically change the playlist that is played depending on the context. It could play "Sue's Italian Cooking" playlist when located in the kitchen and "Chill-out Lounge" in the living room. This behavior is most likely undesirable when music is currently playing and the device is carried from one to another location. But it could be desirable when activated in a new context after being turned off. The identity of the user activating the speaker could be used in a similar way. The physical context could also be used by a digital picture

frame to change the collection of pictures shown. Augmenting those objects with virtual interface as realized by the MIT OpenHybrid Project² would merge context-aware objects with context-aware interface.

Required Technology - An implementation of such object would require a localization service, user identification and activity recognition for the basic functionality. Furthermore the technical component for the actual object such as the music speaker are necessary.

Ambient Information Displays [UC.6]

Functionality - Display of digital information in an ambient manner.

Description - A dynamic art piece could be an implementation for such an application. A landscape painting could transform according to day time and the weather forecast. A family portrait could inform about the presence and activities of all family members. The art piece could also be a kinetic sculpture.

Required Technology - For an implementation of such a system the access to the information that should be displayed is necessary. In case of the landscape painting that would be a data feed from the weather forecast. For the family portrait location and activity tracking of all family members would be required. Of course also the technical components for the realization of the dynamic art piece need to be considered.

²<http://openhybrid.org>

Chapter 4

Design & Development Method

This chapter elaborates on current common design and development methods used in industry and research. Furthermore the impact on and the adaptations necessary to a method for ILEs by incorporating functional ExP will be discussed.

4.1 Current Methods

Development methods based on Agile thinking are nowadays in many cases preferred over e.g. the waterfall model [37]. The concept of agile development originated from the software development community in the late 1980s.

The term *agile* was established with the publication of the *Manifesto for Agile Software Development* in 2001 [7] (see figure 4.1). Over the last years “the integration of user experience design and Agile methods has caught the attention of Agile researchers and practitioners alike” [23]. Jongerius [29, p. 18] state that they first discovered the suitability of an Agile approach implemented in a Scrum method for UX design in 2008. But before diving into the incorporation of UX and User-centered Design (UCD) with such methods, a brief introduction to Agile, Lean and Scrum is given subsequently.

Agile Development Model creates an environment in which requirements and solution evolve through collaboration between self-organizing, multidisciplinary teams in fast iterative cycles. In the earlier mentioned manifesto for Agile development [7] four main values were defined. First, it is important to value individuals and interaction over processes and tools. This can be realized by ensuring an appropriate level of motivation by means of a good work-life balance, enabling self-organizing teams and employees, and fostering interaction by means of e.g. co-location and Pair Programming. Second valued aspect is the delivery of working software rather than a comprehensive documentation. This does not mean that documentation is obsolete. It forms an important part of ensuring that objectives are met. However, the level of documentation needs to be appropriate and the medium used for documentation should match the dynamic nature of the Agile process. Third, end user or client collaboration over the whole project time span is crucial, since requirements can not be fully captured from the very beginning. The concept with its requirements and specifications need to be continuously adapted to match the desired objective in the end. Fourth and last, the project team needs to continuously adapt to

The Agile Manifesto [11]

4 main values

Individuals and interactions

Working software

Customer collaboration

Responding to change

12 Principles

1. Achieving customer satisfaction by early and continuous delivery of valuable software
2. Welcome changing requirements, even in late development
3. Working software is delivered frequently (weeks rather than months)
4. Close, daily cooperation between business people and developers
5. Projects are built around motivated individuals, who should be trusted
6. Face-to-face conversation is the best form of communication (co-location)
7. Working software is the principal measure of progress
8. Sustainable development, able to maintain a constant pace
9. Continuous attention to technical excellence and good design
10. Simplicity — the art of maximizing the amount of work not done—is essential
11. Best architectures, requirements, and designs emerge from self-organizing teams
12. Regularly, the team reflects on how to become more effective, and adjusts accordingly

Figure 4.1: The Agile Manifesto as defined by Beck et al. [7]

changes in short, flexible and iterative loops instead of sticking to a predefined plan and initially predicted milestones. Figure 4.1 give a overview of these 4 values and additional 12 supporting principles for Agile development from the manifesto [7].

Lean Development Model is based on the lean manufacturing method. Poppendieck and Poppendieck [46] adapted those principles for software development. They define seven principles for development as:

1. Eliminate waste - everything that does not add value for the customer should be omitted.
2. Amplify learning - to increase necessary knowledge about the faced desired objective in short iterative cycles in cooperation with the customer instead of generating intensive predictive documents.

3. Decide as late as possible - based on certain facts as a results of learning instead of uncertain assumptions and prediction. Nevertheless, the development process should offer capacity to react to changes.
4. Deliver as fast as possible - so that feedback can be gathered and can be incorporated within the next iteration. A just-in-time ideology can be implemented using tools such as user stories (scenarios) to improve time effort estimation and daily stand-up meetings to foster evaluation and planning of current tasks and interaction between team members.
5. Empower the team - following the guideline “find good people and let them do their own job” management tasks should be encouraging progress, catching errors, and removing impediments and not focus on micro-managing. The developer team needs to be motivated and directed to work towards higher but achievable purposes.
6. Build integrity in - The system needs to deliver a complete, coherent and seamless experience to the customer. This can be achieved by continuous direct customer input and feedback throughout the development process.
7. See the whole - a system is more than the sum of its parts. The interaction between the components needs to be well-defined.

Lean development is closely related to agile development. For this reason, many actual defined development methods implement aspects of both models.

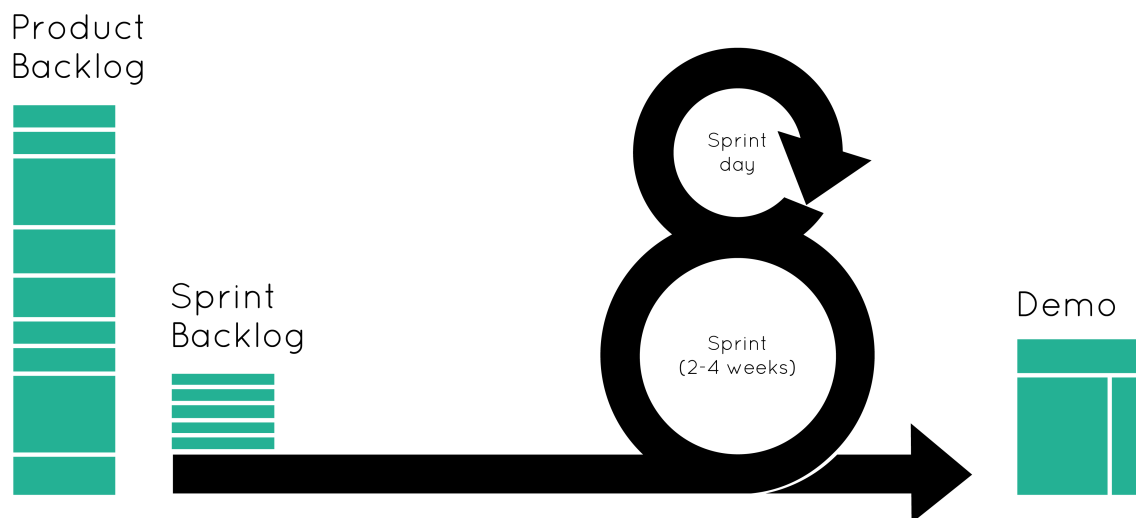


Figure 4.2: One sprint in Scrum based on figure by Jongerius [29]

Scrum is such a method, that implements aspects of Agile and Lean development. During Scrum the stakeholder are divided into different scrum roles within a project [29]. The product owner is a representative on the behalf of the client or customer. He defines the so called user stories and product back logs. The Scrum master handles the daily operation, makes sure that

things such as daily stand-ups are held and is responsible to motivate the team and facilitate the development. The Scrum team is a interdisciplinary team of designers and developers including the product owner, scrum master and other stakeholders. Stakeholders can be any party interested or involved in the project.

Development projects based on Scrum are divided into so called sprints, many iterative development cycles. Figure 4.2 shows an over such a sprint and other important aspects. In the beginning product backlogs, the specifications for the final intended product are defined. Those are split up into more graspable user stories (scenarios). A selection of user stories is made and realized within a sprint within an average duration of two to three weeks. In the end of the sprint outcomes are presented, demonstrated and evaluated. The gathered insights and feedback form the input for the next sprint.

On daily scope so called daily Scrums in form of brief stand-up meetings are typical. The meetings are meant for individual reflection on previous set goals, progress and new tasks to tackle, information exchange on the progress of the overall team as well as fostering collaboration and pointing out necessary cooperation. There is plenty of more in-depth literature on scrum methods available such as by Jongerius [29].

Other Method Implementations that incorporate Agile and/or Lean aspects were developed as well. Kanban enables the management of knowledge with a emphasis on just-in-tim delivery with visual means. An hybrid-method called Scrumban is more suited for maintenance projects or projects with many unexpected tasks due to its flexibility in comparison to the fixed sprint duration in Scrum. Another incremental iterative development method even more focused on the inexpensive creation of high quality source code is called Extreme Programming. In the further course of this work we will however focus on Scrum as underlying design and development method due to its current popularity.

Incorporating UX Design and ExP

The perception of UX in agile literature often implies the separation of UX designers and the developer team [23]. This idea mainly originated from external management rather than from within development teams. A different approach is to add UX designers as team members or to assign a second product owner to ensure that UX aspects are considered during development [23]. Teams should be assembled out of interdisciplinary professionals with different skills and background [48]. Furthermore all stakeholders should be actively involved in all stages of product development. It is important that all involved parties have a common vision on the intended product to ensure efficient work within the design and development team [10]. ExP is powerful technique that mediates experiences and make them sharable within the design team to create a common point-of view. Furthermore, it preserves the subjective dimensions of an experience by active participation of the design team or test user.

UX design nowadays also often “consists of interactive cycles of design, prototyping and validation” [48]. Prototyping support has been an important factor in the design and evaluation of applications [54]. For this reason, several re-

searchers have been working on rapid prototyping tool support for functional [UbiComp] prototypes [54]. Tang et al. [54] define following characteristics of current prototyping procedures for UbiComp applications:

1. Construct prototypes rapidly - since it is inadvisable to spend extensive amount of time on prototyping especially in early design phases.
2. Remove inessential elements - to identify the most important open-design issues and enable better design decision by omitting distracting aspects.
3. Construct prototypes for a particular purpose - since it is potentially "difficult or impossible to create a prototype served in the whole design process" [54].

UbiComp prototypes constructed in this way fulfill according to Tang et al. [54] three different purposes. First, they enable to capture the user's intent for an interaction. Second, to demonstrate the behavior of context-aware applications and third, to test the UX of an system as well as potentially also its technical feasibility. Main purpose of an prototype therefore to gather worthwhile user feedback.

This leads back to the problem motivation of this work as formulated in chapter 2. In-depth research on UX of UbiComp systems is lacking due to technical challenges while prototyping and execution of user studies [55]. Furthermore, user studies need to be executed in real-life context with the real user on long-term to achieve worthwhile results. For example, a main aspect of UX in UbiComp system is learnability, since many system target end users that are novices in use of such technologies. However, learnability can only be evaluated over an extensive period of time. Unfortunately current prototyping tools do not enable the creation of prototypes supporting such studies [54] (also see section 2.4). User-centric design methods ensure the applicability of the final product by the end user in the real Context of Use by continuous close interaction between end user and designer [48].

Another important aspect within a design and development process is the data-gathering method used within user evaluations. As mentioned previously qualitative methods are the most prominent approach in user tests and studies [55]. In the following a short introduction of the most common methods as found by [55] and others is given.

[Eva1.1] Questionnaires can be executed on paper or digitally. They need to take place shortly after the actual user test or more specific the interaction with the prototype. They can aim for quantitative, qualitative as well as a mix of quantitative and qualitative data. They are easy to distribute and standardize but only offer a limited level of details and insights when compared to e.g. interviews.

[Eva1.2] Interviews [35] can take place in one-on-one or group settings. Level of formality can vary also depending whether a structured, semi-structured

or open approach is chosen. Records are most often written transcripts and audio and video recordings. Evaluation efforts of those records can be significantly high depending on the desired level of details and insights.

[Eva1.3] System logging is commonly the gathering of pure quantitative data. The data volume can be high and therefore difficult to be handle manually by an UX expert. In most cases the data needs to be further processed, analyzed and interpreted by an UX expert to extract meaning and derive insights.

[Eva1.4] Observation is either done in direct presence of the user, hidden behind a “magic mirror”, or by live video feed or video recording. The observer transcripts data live and additionally can analyze recording in more detail after the actual evaluation. The gathered information needs to be analyzed and interpreted as well.

[Eva1.5] Diaries and probes [35, p. 209] are long-term studies mainly executed by the participants themselves. They are asked to document e.g. their activities, emotions, impressions, usability and many more aspects in e.g. a diary. Additional a probe can be provided as research material such as prototypes. Close on-going contact between researcher and participants is crucial to ensure correct execution and fast fix of issues such as defect prototypes.

[Eva1.6] Experience sampling and day reconstruction method [16, 30] are variations of a diary study. In comparison to the general concept of diary studies experience sampling specifies and discretizes the point in time for documentation. The day reconstruction method does include the overall day as context to e.g. analyze time-budget management.

[Eva1.7] Community forums & blogs [11] - Offering digital platforms where user can give feedback and ask for help with problems has been a popular approach in industry. For example the music streaming platform Spotify has such a community forum¹ where customers can report bugs, ask for help or propose new features. Employees react to those inquiries e.g. when it comes to the likelihood that a proposed new feature gets implemented in the near future. In this way user feedback can be incorporated during for example an open beta test or after the first release of an application for continuous further development and improvement.

UX experts in industry prefer more informal, significantly shorter but more frequent evaluations in comparison with research [11]. This difference is evident by the nature of the evaluation purpose of the two approaches. UX studies in industry attempt to validate design-choices prior realization efforts and final implementation. UX studies in research attempt to validate scientific findings that are a potential base for future studies and system designs.

¹<https://community.spotify.com/>

4.2 An Optimized Method



Figure 4.4: Presence of expertise within the team.

This section discusses the impact on and the adaption to a design and development method for ILEs caused by incorporating functional ExP.

Figure 4.3 shows an illustration of an optimized design and development method. The main model is based on the Scrum method that implements Agile as well as Lean development principles. We considered furthermore principles for user-centered agile software development as defined by Brhel et al. [9]. The identified 5 principles were incorporated in our design and development method in following manners:

- [OptiM.1] **Separate product discovery and product creation** - An initial zero cycle is used for an clarification of the objective to tackle, ideation, concept generation and validation phase. Only if the initial concept has been validated the development team can enter the creation phase. Otherwise it needs to return to conception or even ideation.
- [OptiM.2] **Iterative and incremental design and development** - The creation phase is structured by iterative incremental cycles consisting each out of design, prototyping and testing. However, our design process attempts to incorporate the concept of Continuous Development as proposed by Kuusinen [36]. Instead of delivering results at the end of a sprint with a fixed duration in Continuous Development results get delivered whenever ready. For this reason, sprints of UX and technical development tasks can have different duration and can be asynchronous. This is caused by e.g. the fact that new UX insights can lead to adaption in the design and therefore lead to necessary changes in the technical development.
- [OptiM.3] **Parallel interwoven creation tracks** - A close, smooth and seamless cooperation between team members is desired within this design and development method. The team should be interdisciplinary in accordance with the Scrum method. UX expertise should be integrated within the team [36]. In our opinion they should be a fluent gradient of expertise between UX and technical development knowledge (see figure 4.4). For this reason, we also prefer to talk about UX and development

task instead of teams. Of course, some team members will be exclusively UX designer and some only developers. However, other team members will have expertise in both fields creating a new type of expert called UX developer. This relation can be translated to combinations of any fields of expertise necessary for the development of ILEs. UX and technical development are only a promoted example here.

- [OptiM.3] **Continuous stakeholder involvement** - The end user is involved already in the end of the conception phase and throughout the whole creation phase. Enabling long-term user studies is one of the main objectives of this work. Running such studies will most likely lead to more valuable insights about the UX of the system in question. A robust functional prototype that provides the same UX as the intended final system is the enabling tool for this method.
- [OptiM.3] **Artifact-mediated communication** - Main objective of this work is to facilitate the time- and cost-efficient functional ExP of ILEs for UX research, design & development. The resulting functional Experience Prototype can be used to mediate the UX of a system to co-workers, stakeholders and end users and forms the Central Design Record (CDR) [9].

The Prototype

In the following some details on the evolution of the prototype throughout the scope of a project as illustrated in figure 4.5 and the bottom of figure 4.3 is given. Low-fidelity Experience Prototypes that can only capture the UX partially are initially created for ideation and conception purposes. Afterwards, previously formulated user scenarios get implemented as functional ExP for the validation of the initial concept. If the concept is validated the functional ExP gets extended to provide the full UX of the intended final system. This enables the start of long-term user studies in an early stage of the creation phase and provides a common understanding of the system to all team members. The functional ExP is built-up from flexible components. Functional Experience Prototype, technical prototype and even final implemented components can fast be interchanged at any point in time throughout the project. In this way the system can grow steadily towards the final implementation in agreement with the Lean principles of eliminating waste and only implementing the minimum necessary. We call this approach Continuous Interdisciplinary Prototyping (CIP). Additionally, the prototype environment provides an improved test setting for the technical development in comparison to current practices.

The cooperation between design and technical field is in this manner supported as well. A rough development flow would progress as following: The flow starts with UX tasks, functional ExP and user studies. Gathered UX insights provide indications which components should be realized as technical prototype next. The technical prototypes are implemented and are tested in the context of the functional Experience Prototype. If successful, they

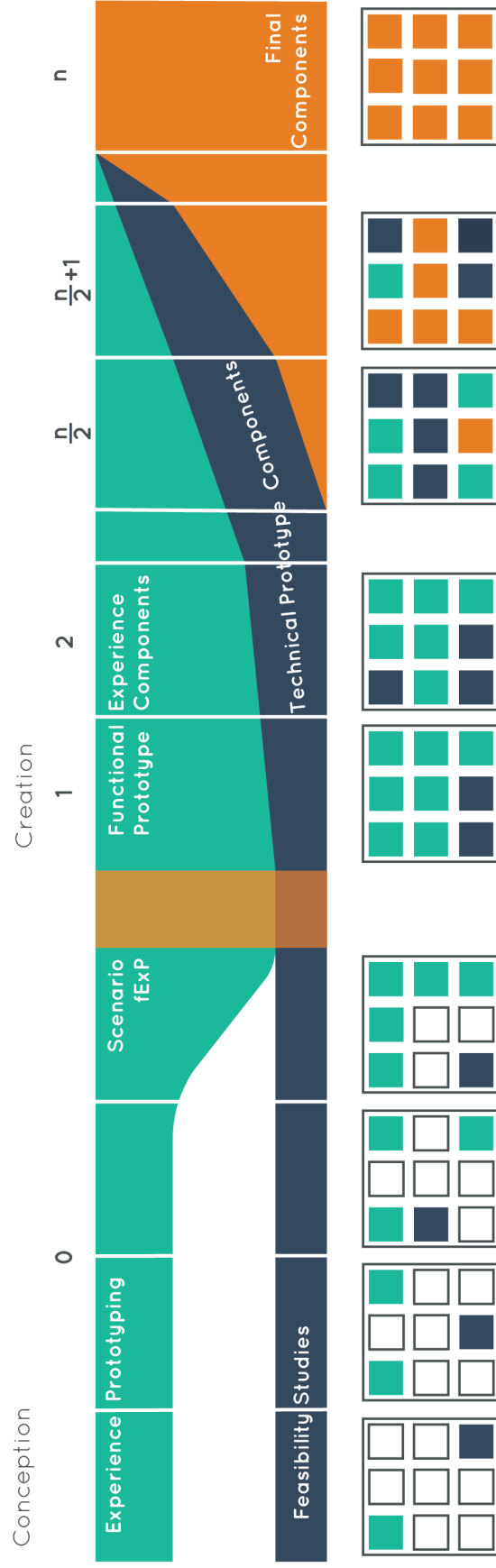


Figure 4.5: Evolution of the prototype through out the design & development process including functional ExP, technical and final components.

enter the stage of development towards a full implemented component. This component can be tested in its various fidelity stages within the functional Experience Prototype as well. Finally it can be deployed within the environment and tested on long-term.

The subsequent section will show an evaluation of the proposed optimized method by professionals in industry.

Evaluation

In this section the evaluation and validation of the applicability of the proposed optimized method is described. First the chosen evaluation method is presented followed by a summary of the execution as well as a short conclusion.

4.2.0.1 Method

The evaluation is carried out in form of interviews with industry experts. They shall preferably have either an expertise in UX/User Interface (UI) design or in the technical development of UbiComp systems. Unfortunately the availability of professionals with expertise within UX design and research for UbiComp systems is limited. Therefore it is potentially necessary to also consider experts with a general expertise in those fields. The interview or discussion will have an informal setting and will be semi-structured. In this way it is ensured on the one hand that we will receive the minimum required information for the evaluation and on the other hand provide opportunity to discuss concepts and aspects in more detail. The interviews were recorded for transcription purposes. In the following the fixed question set in chronological order is listed. The full predefined structure of the interview is available in appendix B.

1. Do you have any question on the concept of functional ExP?
2. What is the typical design and development method used within your organization?
3. What is your opinion on this design and development method for ILEs optimized for functional ExP?
4. Do you think it is realistic and applicable in a real world setting?
5. What points or aspects would you identify as critical or challenging (in a real world setting)?
6. Do you have any other feedback or remarks on the design and development process using fExP?
7. Do you have any other feedback or remarks on this graduation project?
8. Any other question or topic you would like to discuss?

4.2.0.2 Interviews

Several companies were contacted by us about their interest in an cooperation for an evaluation. Fortunately two companies, eFocus based in Utrecht and Nedap located in Groenlo, responded positively. Subsequently, brief summaries of the two interviews carried out are given.

eFocus - received us on May 11 from 15.30 to 17.00 hrs. The interviewee was the Creative Director Floris Ketel, who has an degree in IxD. Furthermore the IxD lead Pieter-



jan Oomen joined halfway through the interview. In general they were really enthusiastic about the concept of functional ExP and the implied adaption to the design and development process. Their answer to the our questions are summarized below.

1. *Do you have any question on the concept of functional ExP?*
No
2. *What is the typical design and development method used within your organisation?*
eFocus uses in general an agile method (Scrum) with sprints with an duration of two weeks in average. A multidisciplinary team defines user stories (product backlogs) in the beginning of project and plans the outcome of each sprint over the whole course of the project. They also have focus session on a regular basis to evaluate past and plan the upcoming sprint. They hardly ever user a waterfall based method anymore. Sometime especially if another company is involved in a project a partial waterfall based approached is chosen.
3. *What is your opinion on this design and development method for ILEs optimized for functional ExP?*
They were enthusiastic about functional ExP and CIP. CIP is really interesting since in this way seamless flow towards final implementation. Functional ExP creates already in early stages a medium to communicate the concept in a very easy understanding graspable way. An interesting aspect they pointed out is what happens after the first final implementation is deployed. Nowadays most system are developed and released in an continuous way. The concept of functional ExP and CIP sounds very good on paper but it would be interesting to see them applied in a real project.
4. *Do you think it is realistic and applicable in a real world setting?*
Yes, given that the toolkit for the support of functional ExP and the design and development method is existing. Furthermore it should be considered what happens when working on an improvement or addition to an existing ecosystem and not a completely new product. When infrastructure is shared between the prototype and deployed services the prototypical components should not be able to have a negative influence

on the deployed system. Another aspect is how insights gathered during the development and user evaluation might change the aimed added value for the end user. Learning about the added value of a whole [environment] (ecosystem) is in some cases a step-wise process component by component. Some IoT devices like the Nike+ sensor integrated in shoes first generated a worthwhile added value in the further development of the ecosystem. First an improved smart phone application made the data collected by the sensor meaningful to the user.

5. *What points or aspects would you identify as critical or challenging (in a real world setting)?*

The seamless replacement of technical prototypes by final implemented components is potentially critical in their opinion based on experiences with deploying software.

6. *Do you have any other feedback or remarks on the design and development process using fExP?*

What is the focus of this design method? Does it consider the added value created for the user?

- *Our Reply:* Yes, it is a UCD method that starts e.g. with the actual user need and continuously involves the user and his feedback, also on the fulfillment of his needs, within the design and development process. Nevertheless, as you pointed out before, some applications are the result of the assembly of IoT products and services that did not have an particular added user value in mind. Therefore, starting with defining added value like our proposed method does is not the only way that should be considered. But certainly a UCD method is most useful if the UX of a product is the main focus of a design and development process.

7. *Do you have any other feedback or remarks on this graduation project?*

They are highly interested in the outcome of this work and also in the functional ExP toolkit and its availability. They will receive a digital copy of this work to express our appreciation for their cooperation after its finalization.

8. *Any other question or topic you would like to discuss?*

Additionally they are interested in cooperation with the University of Twente (HMI, CreaTe) on internships and graduation projects.

Nedap - welcomed us on June 7 from 10.00 to 11.30 hrs. Three employees participated in our evaluation session for our optimized design and development method: Jan Hendrik Croockewit - founder of the healthcare unit, now head of research and development, with an engineering degree in physics; Wouter Kersteman - product developer in the area of security management; and Matthijs Langenberg - software developer at the healthcare unit. The outcome of the session is briefly summarized in the answers below.



1. *Do you have any question on the concept of functional ExP?*
No questions.
2. *What is the typical design and development method used within your organisation?*
The methods differ a lot between units and teams. Nedap is a very technological focused company with mostly business clients (business to business (B2B)). Mr. Croockewit thinks that they in general prototype too less and start right away with building products. Mr. Kersteman remarks that the development of hardware is not as fast iterative as compared to software. Most of the time they make prototypes but do not have enough time to test it with the actual customer or user. Additionally it is also difficult to determine who are the actual user of the product because many parties are touching the product up until it is finally installed. Mr. Langenberg is more involved with software development. In his team they don't create many prototypes but directly create software and start alpha/beta testing with end users such as healthcare staff as soon as possible. For this reason, they integrate channels for user feedback mechanism directly in their software. Within Nedap Agile methods are used in general. Bigger teams use Scrum and smaller teams decide for themselves. During conception they use a 1/10/100 days approach: What should the product look like in 1,10 and 100 days? First feedback comes from business clients and then potential end user get involved.
3. *What is your opinion on this design and development method for ILEs optimized for functional ExP?*
They judge functional ExP as the right way to go for future UX research and design, to spare on programming efforts and prevent throwing away implemented components. The seamless shift from prototype to final implementation seems very attractive to them. They furthermore judge it as highly worthwhile to make the developers empathic or aware of the desired UX of a product. But that has to be continuously done. They agree that user testing should be started as early as possible within the development process. Most likely as consequence the original objective will change.
4. *Do you think it is realistic and applicable in a real world setting?*
No direct answer given.

5. *What points or aspects would you identify as critical or challenging (in a real world setting)?*

It is important to know whether development is happening on an existing product or completely new product. Development on existing products needs to consider the expectations of the existing user base.

6. *Do you have any other feedback or remarks on the design and development process using fExP?*

It is important to define and be aware what is tested during an evaluation so that the outcomes can be measured against the expectations.

7. *Do you have any other feedback or remarks on this graduation project?*

No.

8. *Any other question or topic you would like to discuss?*

No.

The interviews magnified the need of an evaluation of the optimized method during the development of a real product. But unfortunately this is not easily realizable based on multiple reasons. Finding a cooperative development team who is willing to use an unproven tool for the development of a real product is difficult. Furthermore would the development cycle of such a product most likely exceed the time scope of this work. Additionally the resources for the development of an complete functional ExP tool kit are limited as well. Another aspect that became clear during the evaluation is the development on existing products and platforms needs to be more considered since nowadays many products are developed continuously.

But nevertheless, functional ExP was perceived as a valuable method to improve the UX of a product while in parallel minimize development efforts. The CIP that enables a seamless transition from a prototypical to a final product state caught significant interest amongst the interview partner. In general they judge the optimized design method as feasible under the premise that the functional ExP toolkit is available.

Chapter 5

A Functional Experience Prototyping Toolkit

This chapter lays the foundation for a toolkit that enables time- and cost-efficient functional ExP of ILEs. In the first section of this chapter specifications for such a toolkit are defined derived from various knowledge foundations that were created previously in this work. In the subsequent section these specifications are translated into actual tool components that form the formal outline of the toolkit. Afterwards, the implementation of an initial set of tool components is described. The selection for the initial set is derived from the ILE use cases presented previously in section 3.3. Final part of this chapter is a proof of concept of the toolkit in form of functional Experience Prototypes of the selected use cases.

5.1 Specifications

A knowledge foundation has been generated by answering two essential questions previously in this work. First result is an UX outline that provides insights into what kind of UX a functional ExP toolkit for ILEs needs to be able to generate. Second, an optimized design and development process adds on how the toolkit potentially will be used and which functionalities are necessary next to the pure generation of a prototype. Based on the UX outlines for ILEs in chapter 3.1, the optimized design and development method in chapter 4.2 and our own experience in prototyping and developing similar systems the subsequent toolkit specification were defined. The origin of the separate specifications is indicated by identifiers used throughout this work. An overview of these identifiers and references to the associated aspects and definitions is available in appendix A.

Spec.1	... must enable functional ExP of ILEs Origin: [MainObj] Related to: [Spec.2]
Spec.1.1	... must enable prototypical implementation of or connection to all components especially related to implicit interaction defined in figure 3.4. + Origin: [UxOut] → chapter 3.2.3.3 + Related To: [Spec.1.2]
Spec.1.1.1	... must enable the implementation of (prototypical) Input/Output (I/O) hardware.
Spec.1.1.2	... must enable the storage of different kinds of information within an ILE relevant currently and in the past. <i>Such as:</i> <ul style="list-style-type: none"> • User information, activities, routines and habits • Intentions of Use • Context of Use • Devices their status and capabilities • Services • Events and actions + Origin: [TCL.6] + Related to: [I.1-5] [C0.1-5] [Spec.1.1.3]
Spec.1.1.3	... must incorporate the creation of intelligent services for following purposes: <ul style="list-style-type: none"> • Recognition of user and their activity, Context of Use with all its dimensions as defined in chapter 3.2.2. • Learning of user routines, habits and preferences incorporating the Context of Use. • Prediction of user intentions and activities. • Make recommendation on actions, choices and planning to the user concerning the Intentions of Use categories defined in chapter 3.2.1. + Related to: [I.1-5] [C0.1-5] [Spec.1.1.2]
Spec.1.1.4	... must enable the connection to external services such as Application Program Interface (API)s for e.g. weather forecasts, etc.

Spec.1.2	... must enable current means for explicit interaction such as Automatic Speech Recognition (ASR) and speech synthesis. + Origin: [UxOut] → chapter 3.2.3.3 + Related To: [Spec.1.1]
Spec.1.3	... must enable communication between all components including third-party components and remote services. + Origin: [UxOut] → chapter 3.2.3.3, [DesMeth] → to support CIP + Related To: [Spec.1.4]
Spec.1.4	... must enable interoperability between all components including third-party components and remote services. + Origin: [UxOut] → chapter 3.2.3.3, [TCL.2] + Related To: [Spec.1.3]
Spec.1.4.1	... must enable technical interoperability.
Spec.1.4.2	... must enable semantic interoperability.
Spec.1.4.3	... must be built upon standards also used for the development and deployment of ILE applications. + Origin: [DesMeth] → to support CIP + Related to: [Spec.5]
Spec.1.5	... must enable the simulation of devices and events. + Origin: [DesMeth] → to support CIP + Related To: [Spec.5]
Spec.1.5.1	... must enable manual simulation. (<i>e.g. Wizard-of-Oz GUI</i>) + Related to: [Spec.5.1.1]
Spec.1.5.2	... should enable automatic simulation. (<i>e.g. virtual devices</i>) + Related to: [Spec.7]
Spec.2	... must enable time- and cost-efficient functional ExP. Origin: [MainObj] Related to: [Spec.1]
Spec.2.1	... should be at least significant more efficient than the prototypical technical implementation of the system in question.
Spec.2.1.1	except if the prototypical technical implementation is already considered time- and cost-efficient in the context of functional ExP.
Spec.3	... would make functional ExP accessible to technical non-experts. Origin: [DesMeth] [OptiM.3] → Design & Dev Team → Expertise Related to: [Spec.4]
Spec.3.1	... must be easy to use for UX designer/researcher with (web) development skills.
Spec.3.2	... should be easy to use for UX designer/researcher with web design skills.
Spec.3.2	... would be easy to use for any UX designer/researcher within HCI.
Spec.4	... must provide a clear overview over the whole functional ExP. Origin: [DesMeth] [OptiM.2] [OptiM.3] [OptiM.5] Related to: [Spec.3]

Spec.4.1	... must provide an overview over all components within the functional ExP
Spec.4.2	... must provide an overview over all rules learned by the system and defined by an user. + Related To: [Spec.1.1.3]
Spec.4.3	... must provide an overview over all current events (e.g. triggered rules, sensor values, user present, etc.). + Related To: [Spec.4.2] [Spec.4.4]
Spec.4.4	... must provide an overview over all past events. + Related To: [Spec.4.2] [Spec.4.3]
Spec.5	... must support Continuous, Agile and Lean development principles. Origin: [DesMeth] especially [OptiM.2] [OptiM.3] Related to: [Spec.2]
Spec.5.1	... must enable incorporating components with different level of fidelity. + Origin: [DesMeth] → to support CIP + Related To: [Spec.5.2]
Spec.5.1.1	... must enable support or connection to stand-alone Experience Prototypes. (e.g. GUI prototypes) + Origin: [ExP.6–11] + Related to: [OptiM.1] [Spec.1.5.1]
Spec.5.1.2	... must enable connection to Technical Prototypes. + Related to: [Spec.1.4.3]
Spec.5.1.3	... must enable connection to final, already existing and third-party components. + Related to: [Spec.1.4.3]
Spec.5.2	... must enable the exchange of components with different level of fidelity at any point of time. + Related To: [Spec.5.1]
Spec.6	... must support continuous long-term user evaluations. Origin: [DesMeth] especially [OptiM.4]
Spec.6.1	... must enable continuous logging and storage of events within the ILE. + Origin: [Eval.3] + Related To: [Spec.1.1.2] [Spec.4]
Spec.6.2	... should analyze and interpret logged data autonomously. + Origin: [Eval.3] + Related To: [Spec.1.1.2] [Spec.1.1.3] [Spec.4] [Spec.6.1]
Spec.6.3	... should offer simple distribution channels for digital questionnaires and diary studies. + Origin: [Eval.1] [Eval.5] [Eval.6]
Spec.6.4	... should offer simple communication channels to invite selected participants to evaluation sessions. + Origin: [Eval.1] [Eval.2]

Spec.6.5	... would facilitate user evaluation in form of hidden observation. + Origin: [Eval.4]
Spec.6.5.1	... would grant access to live video feeds from within the functional ExP.
Spec.6.5.2	... would grant access to video recordings from within the functional ExP.
Spec.6.6	... must protect the security and privacy of the participants. + Origin: [TCL.6] [IxD.IM.2]
Spec.6.6.1	... must only grant access to user data with permission of the participant in question. + Related to: [Spec.6.1] [Spec.6.2] [Spec.6.5]
Spec.6.6.2	... must store user data in a secure place protected from unauthorized access. + Related to: [Spec.1.1.2] [Spec.6.1] [Spec.6.2] [Spec.6.5]
Spec.7	... must support testing of technical prototypes. Origin: [DesMeth] [OptiM.3] → to support CIP Related to: [Spec.5.1.2]

5.2 A Tool Framework

The specifications defined in the previous section are the essential input to create a toolkit for functional ExP. Nevertheless, these specifications need to be translated into actual tool components. This effort is documented in the subsequent section.

Before we dive into the separate components it is necessary to point out that some of the specifications can not directly be translated into one specific component. For example [Spec.1] and [Spec.2] are in sum basically the main objective [MainObj] and therefore will be realized by the toolkit in its whole. Same is true for [Spec.3], enhanced usability for non-tech-experts, and [Spec.5], the support of common and appropriate development methods. How these specification are implemented by the assemble of tool components will be explained in the end of this section.

In the following a list of tool components divided into the categories prototype management, communication & data storage, I/O, and intelligence enabling services is presented. The implemented specifications and dependencies as well as a short description of the functionality of a component are stated in detail. Figure 5.1 provides an overview over all tool components.

Prototype Management

Envisioned is a central GUI that incorporates the components 1 to 5 and 16 and therefore provides an complete overview as required in [Spec.4].

[Comp.1] Component Management Unit

- Implements: [Spec.4.1] [Spec.4.3] [Spec.4.4] [Spec.5]
- Dependencies: [Comp.6] [Comp.7] [Comp.16]

The *Component Managment Unit* displays all devices and services available. This includes all input and output sources as well as processing units such as the behavior engine [Comp.17] in between. Furthermore the unit should indicate the current status and fidelity level of all components. It should offer the possibility to add new or remove existing components from the prototypical environment. In this way an overview of the whole prototype is available to the whole development team and hence forms a shared knowledge base amongst the team members.

[Comp.2] User Management Unit

- Implements: [Spec.4.1] [IXD.5]
- Dependencies: [Comp.6] [Comp.7]

The *User Managment Unit* must provide an overview over all registered user and their data. Its functionalities include adding and removing users as well as editing their data stored in [Comp.7].

[Comp.3] Data Management Unit

- Implements: [Spec.4.1] [IXD.5]
- Dependencies: [Comp.6] [Comp.7]

The *User Managment Unit* grants access to all the data entered and generated within the prototype environment. It is the access to the virtual representation of the prototype and the documentation of the events happening within. It furthermore must provide the possibility to add, edit or remove data, even though this potentially needs to exclude certain types of data critical for the system.

[Comp.4] Simulation Unit

- Implements: [Spec.1.5]
- Dependencies: [Comp.6] [Comp.17]

The *Simulation Unit* is essential for rapid and CIP. One part of this unit is a tool for the creation of Wizard-of-Oz GUIs. This tool dynamically generates GUIs with buttons, sliders and other interface elements on demand. These interfaces are used to simulate inputs, outputs and intelligence in early user tests or for technical testing of prototype components. A second functionality shall facilitate this as well by enabling the simulation of prototype components that are absent or not yet ready. This could potentially be realized with limited efforts by a secondary interface to the *Behavior Engine* ([Comp.17]) that only contains rules that represent a virtual device.

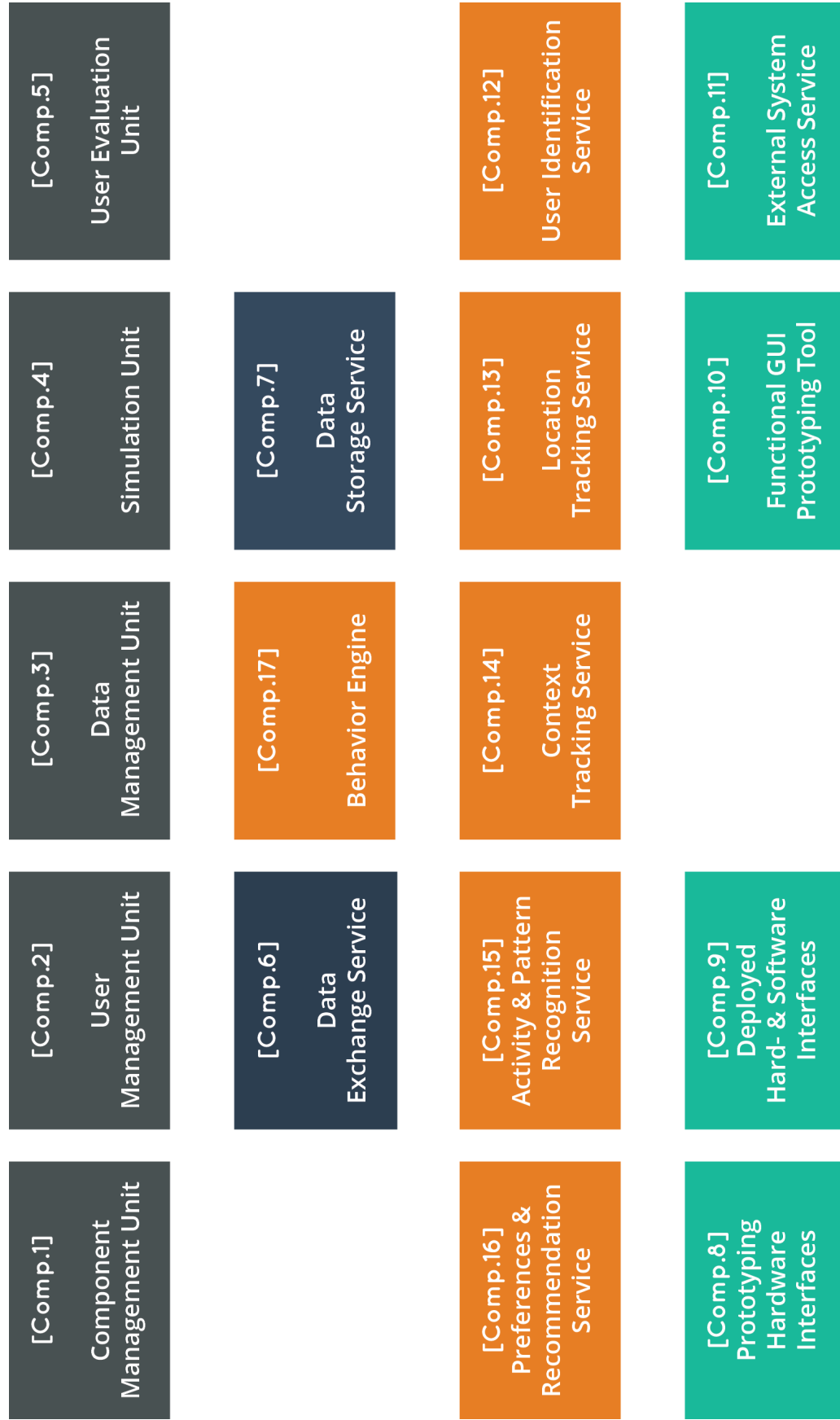


Figure 5.1: Overview over all components of the functional EXP toolkit.

[Comp.5] User Evaluation Unit

- Implements: [Spec.6]
- Dependencies: [Comp.6] [Comp.7]

This unit must support continuous user evaluation throughout the whole design and development process. In cooperation with [Comp.7] it must grant access to specified data logged throughout the use of the system. Additionally it would be beneficial considering the intelligent capabilities of the toolkit ([Comp.12–17]) if the data would at least partially be interpreted autonomously. Furthermore it should provide the possibility to manage multiple user evaluations and the corresponding participants based on the information in [Comp.2]. The *User Evaluation Unit* should include functionalities to easily distribute questionnaires, executed diary studies and send invitations for on-location sessions such as interviews to participants.

Communication & Data Storage

The two components for data exchange and storage are closely related. They need to be built on an agreed *knowledge representation*. An agreement on schematics for storing and exchanging data on users, devices, services, etc. is essential.

[Comp.6] Data Exchange Service

- Implements: [Spec.1.3] [Spec.1.4] [Spec.5]
- Dependencies: basically all components but especially [Comp.7]

The *Data Exchange Service* is one of the most crucial components of the toolkit. It connects all prototype components smoothly and ensures continuous exchange of information and smooth interaction flows between them. Important is that this service facilitates the seamless integration and dynamic exchange of prototype components with different fidelity levels. Therefore it must be build upon industry standards such as presented in chapter 2.1.

[Comp.7] Data Storage Service

- Implements: [Spec.1.1.2]
- Dependencies: [Comp.1–5] [Comp.8–10] [Comp.12–17]

The *Data Storage Service* is available to all other components via [Comp.6]. It stores data about everything that is contained and happening within the prototype environment. The data needs to be structured in predefined schemas known to all components so that they can understand and use the data. The stored data is accessible for users via [Comp.1], [Comp.2], [Comp.3], [Comp.5] and [Comp.17].

I/O

These components assemble the perception layer (see figure 2.3) of the system that interacts with the end user.

[Comp.8] Prototyping Hardware Interfaces

- Implements: [Spec.1.1.1] [Spec.5.1] [Spec.5.1.1] [Spec.5.1.2]
- Dependencies: [Comp.6], hardware and matching firmware and APIs

These interfaces must offer the possibility to easily add and configure prototypical hardware components to the functional Experience Prototype. The software configuration of the hardware as well as the establishment of the connection to other prototype components needs to be easy and fast. An overview of popular prototyping hardware used within research and industry is available in figure 2.5 in chapter 2.1. At least a major share of these should be supported as well as other prototyping platforms aimed on more novice users such as littleBits¹.

[Comp.9] Deployed Hardware & Software Interfaces

- Implements: [Spec.1.1.2] [Spec.1.3] [Spec.1.4] [Spec.5.1.3]
- Dependencies: [Comp.3] [Comp.6], hardware and matching APIs and Software Development Kit (SDK)s

Next to interfaces to prototypical hardware the toolkit must support the incorporation of existing deployed hard- and software solutions. This enables amongst others the use of current means for explicit interaction. This unit should include interfaces to hardware such as the Microsoft Kinect², Leap Motion³ or Myo Wristband⁴. But also for multipurpose hardware like smartphones or software solutions for e.g. ASR and speech synthesis, Instant Messaging Bots (IM bot)s, and GUIs.

[Comp.10] Functional GUI Prototyping Tool

- Implements: [Spec.5.1.1]
- Dependencies: [Comp.6] [Comp.7], GUI prototyping tools

A tool for functional GUI prototyping supports CIP. We understand functional GUI prototyping as the extension of the functionalities of clickable and animated GUI prototypes. They get connected to a remote backend that is connected to other components and simulates the desired behavior and interaction. In this way high fidelity GUI prototypes created with tools like Framer, Axure, UXPin or Adobe XD can be integrated into a functional Experience Prototype from an early stage as placeholder for the actual final application.

¹<http://littlebits.cc/>

²<https://developer.microsoft.com/en-us/windows/kinect>

³<https://www.leapmotion.com/>

⁴<https://www.myo.com/>

[Comp.11] External System Access Service

- Implements: [Spec.1.1.4]
- Dependencies: [Comp.6],

Context-aware application and environment such as ILEs require next to internal data also information from external sources such as weather forecasts, calendar events, traffic information, etc. The *External System Access Service* represents a central access point to such remote services to all components within the ILE prototype.

Intelligence Enabling Services

The differences between a pure connected or smart application, in our opinion, is that smart applications act on the gathered and provided data intelligently to pursue potentially common objectives. Creating such applications and prototypes of such requires high efforts. The components described subsequently should simplify the fast and easy creation of prototypical context-aware intelligent behavior.

[Comp.12] User Identification Service

- Implements: [Spec.1.1.3]
- Dependencies: [Comp.6] [Comp.7] [Comp.13]

The *User Identification Service* uses data from various resources such as the location of personal belongings as for instance a smart watch or phone or facial recognition in security camera feeds to identify users. Each resource associated to a user contributes to a confidence score representing the likelihood that the user is identified correctly. This information can be used by applications for easy access or personalization of interaction but is also used by [Comp.13] for user localization.

[Comp.13] Location Tracking Service

- Implements: [Spec.1.1.3]
- Dependencies: [Comp.6] [Comp.7] [Comp.12]

The *Location Tracking Service* determines the spacial position of mobile objects and users by information gathered by immobile devices such as BLE scanners, NFC reader and cameras. It determines the approximate position of an object and sends it via [Comp.6] to [Comp.7]. Through the continuous storage a location history of every object is generated and is available to all other components for e.g. further analysis.

[Comp.14] Context Tracking Service

- Implements: [Spec.1.1.3]
- Dependencies: [Comp.6] [Comp.7] [Comp.12] [Comp.13] [Comp.15] [Comp.17]

The *Location Tracking Service* lays the foundation to keep track of the *Context of Use*. After determining the physical context [C0.3] all other context information can be determined (see chapter 3.2.2) such as surrounding people [C0.2] and devices [C0.4], day time [C0.1] [Comp.11], user activities [C0.5] [Comp.15], etc. The context information can then be used for simplifying interaction and generate context-aware intelligent behavior ([Comp.17]). On long-term this information can be used for future classification tasks as executed by [Comp.15],

[Comp.15] Activity & Pattern Recognition Service

- Implements: [Spec.1.1.3]
- Dependencies: [Comp.6] [Comp.7] [Comp.12] [Comp.13] [Comp.14] [Comp.16] [Comp.17]

Based on the current and past identified *Context of Use* activities of the user and his daily routines can be recognized. However, the *Context of Use* and the activity, habits and preferences ([Comp.16]) of a user are interrelated and influence each other. Therefore, a recognized activity can increase the precision of the recognized *Context of Use* and vice versa. Correct activity recognition supports the generation of context-aware intelligent behavior ([Comp.17]) and identifying daily routines and habits potentially increases the prediction accuracy of future activities ([Comp.15]).

[Comp.16] Preference & Recommendation Service

- Implements: [Spec.1.1.3]
- Dependencies: [Comp.6] [Comp.7] [Comp.12] [Comp.13] [Comp.14] [Comp.15] [Comp.17]

Stored data from input devices and the associated *Context of Use* can also be used to identify preferences of the user. This learned data can be used to simplify interaction or make recommendations for similar choices or interests in the future. For example an interface could directly offer the functionalities a user prefers in the specific *Context of Use* he is in. Offering well-founded recommendations to the user becomes significantly important with a steadily increasing amount of choices available and independent self-determination of the consumer.

[Comp.17] Behavior Engine

- Implements: [Spec.4.3]
- Dependencies: [Comp.6] [Comp.7] [Comp.12] [Comp.13] [Comp.14] [Comp.15] [Comp.16]

By analyzing and learning on the data gathered by [Comp.12–16] the *Behavior Engine* should create new behavioral rules for an ILE to fulfill specific or more general *Intentions of Use* (see chapter 3.2.1). The engine should furthermore provide an overview of all rules generated or added by the user manually. This overview should also indicate which rules have been triggered currently or in the recent past. The tool component must include the possibility to add, edit or remove rules by the user. How to represent the rules, especially the ones learned by the system itself, has yet to be determined. Possible manners for rule representations could be in IFTTT⁵, Google Blockly⁶, natural language or in source code.

Overarching Specifications

Some of the tool specifications are implemented by a combination of multiple tool components. An agile, continuous and lean development process ([Spec.5]) is supported by the toolkit especially by means of the prototype management components [Comp.1–5] and data exchange service [Comp.6]. The usability of the toolkit even for non-developers is favored by the simple and ready-to-use tool components for I/O [Comp.8–11] and intelligence [Comp.12–17]. However, the actual implementation of the tool components and the corresponding user interfaces are the main influence on the fulfillment of this specification ([Spec.3]). The main objective ([MainObj]) to facilitate time- and cost-efficient functional ExP ([Spec.1][Spec.2]) is realized by tool components that enable the rapid creation of an ILE prototype ([Comp.6–17]) and providing appropriate tools to manage and smoothly integrate the prototype into the design and development process ([Comp.1–6]).

5.3 Implementation

This section deals with our underlying thoughts and a proof of concept for a future implementation of the functional ExP for ILEs. An implementation of all toolkit components additionally to the generated knowledge foundation and derived specifications would unfortunately exceed the scope of this work. Subsequently, first some general implementation concepts and the potential incorporation of existing platforms are elaborated. This elaboration is followed by the description of a proof of concept implementation of the tool guided by two of the use cases presented earlier in chapter 3.3.

In IoT and Smart Environments hard- and software are highly distributed [CR.3]. Tasks get divided onto multiple devices to e.g. cope with limited computation capabilities or increase efficiency. However, for rapid prototyping of distributed systems a centralized approach is potentially more beneficial. Consolidating device management and configuration, communication and system behavior in one place evidentially decreases complexity and required prototyping efforts. However this architecture is impractical for many real life contexts.

⁵<https://ifttt.com/>

⁶<https://developers.google.com/blockly/>

The complexity of many real life systems potentially exceeds the capabilities of a centralized architecture. Therefore functional Experience Prototypes of an ILE will most likely not end up being a final implementation.

Due to the high level of connectivity within IoT and Smart Environments web technologies gained popularity for the implementation of such systems. In agreement with this trend we choose for JavaScript and NodeJS⁷ as underlying development stack. The advantage of NodeJS is its steadily growing open source community and the consequential high amount of libraries for e.g. APIs or devices available such as for the Myo Wristband.

Also other developers built upon the advantages of NodeJS as development framework to create prototyping platforms for UbiComp and IoT applications. The following subsection presents two of these platforms.

Building Upon Existing Platforms

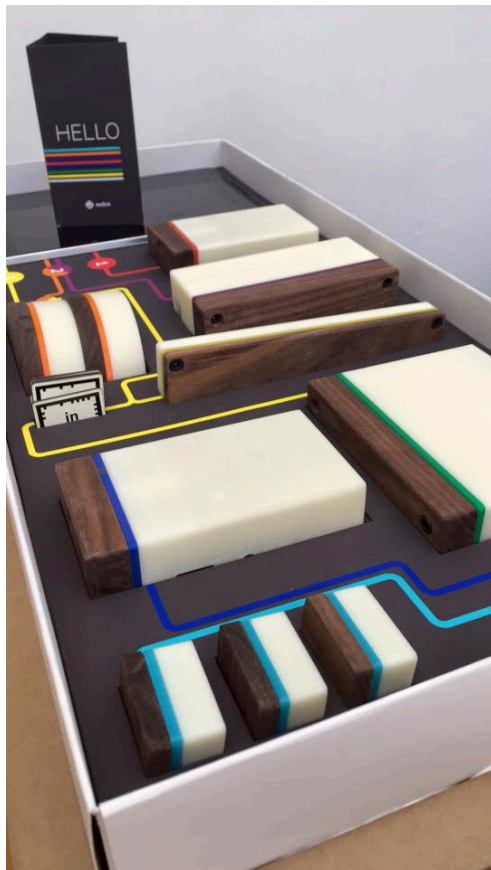


Figure 5.2: Hardware of the meSch kit.

Source: [34, 58]

software is available for various hardware such as Raspberry Pi, Arduino, Intel

In earlier works and an extensive preparatory study we used and investigated current developing technologies for IoT. This also included two prototyping platforms that partially already fulfill our specifications for a functional ExP toolkit. A logical consequence is to build upon and integrate these platforms within our implementation. Subsequently, a short introduction to the two platforms, the meSch kit and Node-RED, that are also based on NodeJS is given.

meSch Project

The project on “Material encounters with Digital Cultural Heritage” (meSch)⁸ is funded by the European Union. Its objective is to enable curators to create interactive tangible exhibits themselves. As part of this project the meSch kit was created (see figure 5.2) [34, 58].

It also follows a centralized architecture. Central element of the kit is a server on a Raspberry PI with WiFi and BLE connectivity called meschHub. After installing the meSch client software on a device, it can be managed and configured via an web interface on the server ([Comp.1]). Client

⁷<https://nodejs.org/>

⁸<http://mesch-project.eu/>

Edison and Android ([Comp.8] [Comp.9]). The server also includes a behavior engine where rules can be defined in JavaScript snippets (partially [Comp.17]). These snippets can get different execution priorities assigned and can be arranged in groups.

The editor for the creation of the rules was made with the end users, the curators, in mind. For simplifying the programming activity two features were conceptualized and implemented. First feature is an autocomplete functionality that assists to find e.g. the desired sensor value of a device. Second feature is something called live-programming. After finding the correct sensor with the autocomplete functionality live values of the sensor can be requested and even the minimal, maximal and average value are calculated. This feature also allows to directly access the last triggered event and its value via a keyboard shortcut. Their aim in usability for curators is comparable to our specification [Spec.3].

Fortunately we were granted the opportunity to work with the platform that is not yet open to the public for this work.

Node-RED

Node-RED “is a visual tool for wiring the Internet of Things.”⁹ It was initially created in 2013 by IBM Emerging Technology lead by Nick O’Leary and Dave Conway-Jones¹⁰. It provides a browser-based flow editor in which nodes that represent software functionalities or connections to hardware and APIs can be assembled to applications. Furthermore it provides an easy way to connect to services via application protocols popular for IoT implementations such as MQTT or UDP. The high amount of modules for NodeJS makes it easy to extend the available nodes with new APIs ([Comp.11]). Some of the nodes for Node-Red also implement components of our toolkit. For example the dynamic UI builder contributed by Andrei Tatar¹¹ would be an excellent Wizard-of-Oz interface creator ([Comp.4]). Node-RED allows additionally to easily build small back-end applications that simulate simple behavior of devices.

Even though fortunately these platforms are already existing and offering great opportunities they still lack components that are essential for our toolkit. But both platforms have not been conceptualized for the the creation of an prototype of an ILE. Tool components that provide a complete overview of the prototype and components that enable context-aware and intelligent behavior are missing. The next subsection presents the implementation of some of these components necessary for the realization of specific use cases. How the before mentioned platforms contributed to the realization of the tool components is also mentioned in more detail.

⁹<http://nodered.org/>

¹⁰<https://github.com/node-red/node-red>

¹¹<https://www.npmjs.com/package/node-red-contrib-ui>

Proof of Concept

In this section a proof of concept implementation of a limited set of tool components is described. The set of tool components was chosen using two of the use cases presented in chapter 3.3 as guideline. We chose for the use cases [UC.1] and [UC.2]. In the subsequent subsections a short description of the implementation of each of the components is given.

meSchHub

Devices

Events

Behaviour

More

Plugins

Settings


New Group

New Device

Filter Groups

Manage Groups

Ping All

 Device Pool

Configured



















State and last activity	Signal	Type	Name	UUID	Device Groups	Edit Device
<div><div>● Pause</div><div>2 days ago</div></div>	2 <div>0 ●</div>	<div></div> android_moto_g	8fb5507e06c7c98	SmartphonesBLE Scanner	<div></div> Configure	
<div><div>● Pause</div><div>2 days ago</div></div>	2 <div>0 ●</div>	<div></div> android_htc_onemini	84:7a:88:24:b8:83	Smartphones	<div></div> Configure	
<div><div>● Pause</div><div>14 minutes ago</div></div>	14 <div>0 ●</div>	<div></div> PI_001	74:da:38:6b:06:c3	BLE Scanner	<div></div> Configure	
<div><div>● Pause</div><div>14 minutes ago</div></div>	14 <div>0 ●</div>	<div></div> PI_000	74:da:38:62:2f:49	BLE Scanner	<div></div> Configure	
<div><div>● Pause</div><div>2 days ago</div></div>	2 <div>0 ●</div>	<div></div> bean_01	d03972ec33e3	BLE Beacons	<div></div> Configure	
<div><div>● Pause</div><div>2 days ago</div></div>	2 <div>0 ●</div>	<div></div> android_samsung_s6	EC:9B:F3:92:50:AB	SmartphonesBLE Scanner	<div></div> Configure	
<div><div>● Pause</div><div>8 minutes ago</div></div>	8 <div>33 ●</div>	<div></div> NotificationModule	c595fcc8d5c	BLE Beacons	<div></div> Configure	
<div><div>● Pause</div><div>12 minutes ago</div></div>	12 <div>44 ●</div>	<div></div> Blidget_Black	f929270d96c2	BLE Beacons	<div></div> Configure	
<div><div>● Pause</div><div>2 days ago</div></div>	2 <div>0 ●</div>	<div></div> PI_002	74:da:38:6b:06:c3	BLE Scanner	<div></div> Configure	

Figure 5.3: Device pool overview in the front end of the meSchHub

[Comp.1] Component Overview Unit

The *Component Overview Unit* is essential to enable the assembly of all necessary ILE components into functional ExP of the use cases. As described before it offers the possibility to add and remove ILE components from the environment. Therefore this tool component would be crucial for any use case implementation.

The meSchHub already offers a similar overview in its web browser interface. It shows two separate lists of all configured and unconfigured devices. Unconfigured devices can be added to the prototype and configured devices can be removed. As soon as the meSch client software is installed on a device it appears in the list of unconfigured devices. Figure 5.3 shows a screenshot of the meSchHub GUI.

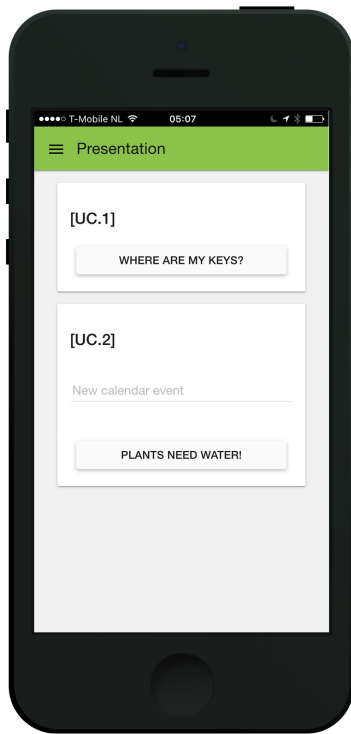


Figure 5.4: GUI generated with Node-RED.

Nevertheless, this GUI does not completely fulfill our vision for this tool component. It lacks the ability to display the fidelity level of the different components, current events and the connection between individual components.

[Comp.4] Simulation Unit

This unit is not crucial for the chosen use cases. However, since it is already implemented as contribution to Node-RED we decided to include it in this list. Figure 5.4 shows a screen shot of an interface build with the library *node-red-contrib-ui*¹².

[Comp.6] Data Exchange Service

In chapter 2.1 some popular IoT application protocols were introduced. Based on previous experiences we chose for MQTT as application protocol to exchange data between tool and prototype components. The publish/subscribe mechanics are beneficial since they enable dynamic n-to-n connections as well as the dynamic exchange of components. Client subscribe to topics on a broker that receives messages from other client that published to the corresponding topic. It is unimportant for the subscribed client if the client that publishes to the topic gets replaced. In this way a functional ExP component can be easily exchanged by a technical prototype or simulated by a Wizard-of-Oz GUI that just sends a message on button press. We chose for the open source Mosquitto¹³ MQTT broker and installed it on the same Raspberry Pi as the meSchHub server, so that it has a fixed Internet Protocol (IP) address within the local network. Client software is available for multiple platforms and programming languages such as Arduino, JavaScript, C++ and more.

[Comp.7] Data Storage Service

For the *Data Storage Service* we chose for a MongoDB instance as database. By means of a third party library for Node-RED we created rapidly a interface to MQTT. In this way all components can request and store data easily by publishing messages to two different MQTT topics. Important for this to work is an agreed schematic to sent and store data. Our preliminary studies have shown that semantic interoperability efforts are lagging behind but frameworks like OWL or Google's Weave protocol are existing (see chapter 2.1). In the future efforts on semantic interoperability should be promoted, since it lays the foundation for intelligent applications.

¹²<https://www.npmjs.com/package/node-red-contrib-ui>

¹³<http://mosquitto.org/>

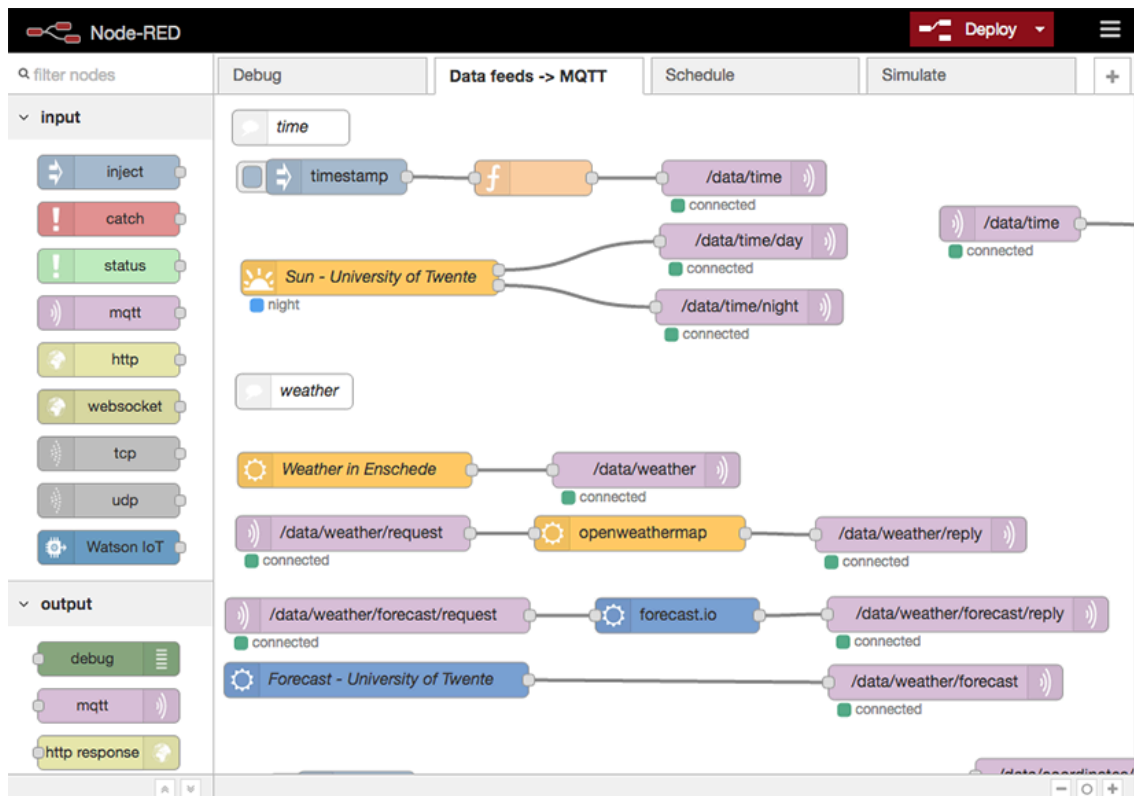


Figure 5.5: Screenshot of the several API nodes connected to MQTT topics in Node-RED.

[Comp.8] Prototyping Hardware Interfaces &

[Comp.9] Deployed Hard- & Software Interfaces

The meSch platform already provides clients for Linux, Windows and Android based systems which includes PC, Raspberry Pis, Intel Edison, Arduino and their own BLE platform Blidgets¹⁴. Once the client is installed the hardware can be configured with different modules such as touch or speech input, a web display, a BLE scanner, or a interface for other hardware such as sensors.

[Comp.11] External System Access Service

We rapidly implemented a MQTT interface to various APIs by means of Node-RED. Figure 5.5 shows how easily installed nodes for weather forecasts, sunrise and -set, Google Calendar, Maps, and Places and more can get connected to a MQTT broker. Node-RED nodes are available for many services and hardware such as the Pebble smart watch, Fitbit fitness wristband, or Nest Thermostat; services such as Dropbox, Slack, Trello, or Amazon's Alexa, libraries for Neural Networks, databases, communication protocols or speech synthesis; and many more. Due to the steadily growing amount of NodeJS libraries the amount of nodes is easily extendable.

¹⁴<http://blidget.hcilab.org/>

[Comp.13] Location Tracking Service

The final implementation will be a NodeJS application that runs in the background and processes the BLE beacons identified by the BLE scanner modules within the meSch platform. The application then decides to which BLE scanner the BLE beacon was closest in proximity. The BLE beacon subsequently has to be associated with the correct person or object by requesting a look up from [Comp.7]. Unfortunately for now data can only be entered into the meSch platform via MQTT but not send outside of the platform. Therefore temporarily the internal global variable storage of the meSchHub is used to store the required data. But the storage capacities are more limited then the intended final implementation and the data is only available as long as the server is running. After a restart the stored information is lost.

Assembly: Extended Memory [UC.1]

For realizing [UC.1] first the meSch client software has to be installed on three Raspberry Pis ([Comp.8]) and one Android smartphone ([Comp.9]). The Raspberry Pis get configured as BLE scanner, the Android smartphone as speech input module and one Raspberry Pi as a web display module ([Comp.1]). Several objects such as a key chain and glasses get equipped with a BLE beacon. The BLE scanner and beacons get associated with the objects and locations. We use the Behavior Engine of the meSchHub ([Comp.17]) to write a small script that keeps track of the location of the user ([Comp.13]). Another script handles the interaction with the user by processing the speech input such as "Where are my glasses?". It looks up the associate beacon ID and the location where it was last seen by the system. Then it displays the result of the query in a natural language text such as "Your glasses are in the living room." on the web display.

Assembly: Context-aware Reminders [UC.2]

For this realization of the use case we do not aim for the most sophisticated scenario that includes user localization. In this scenario we want to send context-triggered reminders to the user as smartphone notification or to a personal notification device. To achieve this the meSch client software needs to be installed on a Android smartphone ([Comp.9]) and a Blidget notification module ([Comp.13]) that incorporates light, sound and tactile feedback. As described before we created a MQTT interfaces to various APIs in Node-RED. Partially those interfaces request new data automatically in fixed frequencies. A script in the Behavior Engine of the meSchHub will react to those sent information and sent out reminders to the user to e.g. water the plants because it has not rained in two days. This meSch Behavior Engine is certainly not the [Comp.17] that we envision but already forms a decent starts with a lot of possibilities.

These two examples show that it is already possible to create functional ExP of ILE use cases even with a limited tool set. In the following chapter the outcomes and findings of this work will be discussed in more detail.

Chapter 6

Discussion & Conclusion

This work attempted to facilitate the time- and cost-efficient functional ExP of ILEs, which resulted in a formal description of an functional ExP toolkit. However, two underlying questions had to be clarified first to tackle this problem statement. The desired UX within ILEs that is supposed to be prototyped and the impact and adaption on a design and development process for ILEs caused by introducing functional ExP had to be determined.

First result was a generic UX outline of ILEs with three dimensions: the Intentions of Use, Context of Use and the characteristics of the system in question. These system characteristics include specific aspects for UX and IxD within ILEs and a set of abstract components that ILEs can be assembled from. For illustration purposes furthermore a set of ILE use case examples was created. They mediate our vision on future ILE applications and are also used as input and proof of concept for the functional ExP toolkit. This UX outline forms furthermore an excellent guideline for the design and development of ILEs. Second result is an optimized ILE design and development process with functional ExP. Current methods were elaborated and advice from current research for the incorporation of UCD in an Agile and Lean development process had been considered. The result is an agile, lean and continuous development process for ILEs. The mix out of functional ExP and Continuous Development principles enables Continuous Interdisciplinary Prototyping. An evaluation by industry professional from corresponding fields has indicated that the concept of functional ExP and the optimized design and development method with CIP is perceived as promising and interesting to pursue. A more comprehensive evaluation of the optimized design and development method and the functional ExP toolkit in a real world setting would have been evidentially more significant in meaning. But both, the implementation of the complete functional ExP toolkit as well as an application in a real project would have exceeded the scope of this graduation project. Main contribution of this work is therefore the derived toolkit specifications, the translation into actual toolkit components and the start on the toolkit in form of a proof of concept implementation. The proof of concept has shown that already with a limited tool set the prototyping efforts for ILE use cases can be significantly decreased.

In the beginning of this work the focus was narrowed from Smart Environment to ILEs, justified by the high dependence of UX on the user, his Intention and the Context of Use, and the system characteristics. Interesting is whether at least part of our findings and outcomes can be applied to the broader

scope of UX in Smart Environment as well. The underlying UX dimensions would be retained since they are domain-independent. The Intentions of Use would certainly need to be extended. However, since ILEs were defined as a subset of Smart Environments all of the Intentions of Use found in this work also apply to Smart Environments. Same is true for the Context of Use and the system components but not for the specific aspects for UX and IxD. These would potentially be exactly the same due to the fact that they are based on general IoT system characteristics.

In this work certain digital prototyping methods were neglected when evaluating current available ExP methods. Kuniavsky [35] and Tang et al. [54] elaborate both on Digital Simulation Environments such as UbiWise [6] and UbiReal [45]. The use of Virtual Reality (VR), Augmented Reality (AR) and mixed Reality becomes especially interesting with new VR and AR headsets entering the market. However, in our opinion, those methods still miss the physical and spacial aspects of a real prototype which are important factors for implicit interaction.

Another aspect that was mostly neglected are prototyping activities concerning ergonomics and aesthetics of physical objects. As stated before, physical aspects of a system are highly important for ILEs, IoT and similar systems and are potentially major influencing factors on the interaction with an object. Which aspects of physical modeling are worthwhile to concentrate on and other perspectives for future work are explained in the concluding chapter of this work.

Chapter 7

Future Work

Industrial Design becomes more important for IT especially in a world where every day things get connected to the internet and become smart devices. In our opinion high potential for future research is in the rapid creation of a technical prototype that is already integrated in a physical prototype of the final object. Technologies such as mobile 3D scanning and 3D modeling could be used to quickly generate a high-fidelity casing for hardware components. The hardware components and handcrafted physical model could be scanned and integrated. An alternative would be a 3D model completely created within a 3D modeling program. In our opinion a software that could arrange the hardware components in such a way to fit the inside of a 3D Model of an object automatically would be a highly valuable tool.

There is certainly high potential for AR and mixed Reality within Smart Environment applications. However, in our point of view its application domain is less development and prototyping but more end user applications such as the dynamic augmentation of environments and objects around the end user.

Concluding we would like to emphasize once more that this work is meant as an initiator for a toolkit for functional ExP. High efforts are still needed after the finalization of this project to make it a valuable toolkit for UX design and research. As soon as all technical components used within the proof of concept are available under open source license, the outcome of this work will be published as open source project, too. After finalization of a major part of the toolkit components an actual long-term evaluation in a real world context can be reconsidered.

Bibliography

- [1] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash. Internet of things: A survey on enabling technologies, protocols, and applications. *Communications Surveys & Tutorials, IEEE*, 17(4):2347–2376, 2015.
- [2] L. Alben. Defining the criteria for effective interaction design. *interactions*, 3(3):11–15, 1996.
- [3] A. Anvari-Moghaddam, H. Monsef, and A. Rahimi-Kian. Optimal smart home energy management considering energy saving and a comfortable lifestyle. *Smart Grid, IEEE Transactions on*, 6(1):324–332, 2015.
- [4] K. Ashton. That ‘internet of things’ thing - in the real world, things matter more than ideas., 2009. <http://www.rfidjournal.com/articles/view?4986>.
- [5] L. Atzori, A. Iera, and G. Morabito. The internet of things: A survey. *Computer networks*, 54(15):2787–2805, 2010.
- [6] J. J. Barton and V. Vijayaraghavan. Ubiwise, a simulator for ubiquitous computing systems design. *Hewlett-Packard Laboratories Palo Alto, â AI HPL-2003-93*, 2003.
- [7] K. Beck, M. Beedle, A. Van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J. Highsmith, A. Hunt, R. Jeffries, et al. The agile manifesto, 2001. <http://www.agilemanifesto.org/>.
- [8] M. Brereton, A. Soro, K. Vaisutis, and P. Roe. The messaging kettle: Prototyping connection over a distance between adult children and older parents. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 713–716. ACM, 2015.
- [9] M. Brhel, H. Meth, and A. Maedcher. Exploring principles of user-centered agile software development: A. *Information and Software Technology*, 61(C):163–181, 2015.
- [10] M. Buchenau and J. F. Suri. Experience prototyping. In *Proceedings of the 3rd conference on Designing interactive systems: processes, practices, methods, and techniques*, pages 424–433. ACM, 2000.
- [11] Å. Cajander, M. Larusdottir, and J. Gulliksen. Existing but not explicit-the user perspective in scrum projects in practice. In *Human-Computer Interaction-INTERACT 2013*, pages 762–779. Springer, 2013.
- [12] D. Clark. Smart-home gadgets still a hard sell, January 2015. <http://www.wsj.com/articles/smart-home-gadgets-still-a-hard-sell-1451970061>.

- [13] M. Clark and P. Dutta. The haunted house: Networking smart homes to enable casual long-distance social interactions. In *Proceedings of the 2015 International Workshop on Internet of Things towards Applications*, pages 23–28. ACM, 2015.
- [14] C. Coelho, D. Coelho, and M. Wolf. An iot smart home architecture for long-term care of people with special needs. In *Internet of Things (WF-IoT), 2015 IEEE 2nd World Forum on*, pages 626–627. IEEE, 2015.
- [15] Y. P. Cruz, C. A. Collazos, and T. Granollers. The thin red line between usability and user experiences. In *Proceedings of the XVI International Conference on Human Computer Interaction*, page 46. ACM, 2015.
- [16] M. Csikszentmihalyi and R. Larson. Validity and reliability of the experience-sampling method. In *Flow and the Foundations of Positive Psychology*, pages 35–54. Springer, 2014.
- [17] J. Gantz and D. Reinsel. The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east. *IDC iView: IDC Analyze the future*, 2007:1–16, 2012.
- [18] J. J. Garrett. The elements of user experience. *User-centered design for the web*. United States of America, 2002.
- [19] R. Gervais, J. Frey, A. Gay, F. Lotte, and M. Hachet. Tobe: Tangible out-of-body experience. In *Proceedings of the TEI '16: Tenth International Conference on Tangible, Embedded, and Embodied Interaction*, TEI '16, pages 227–235, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-3582-9. doi: 10.1145/2839462.2839486. URL <http://doi.acm.org/10.1145/2839462.2839486>.
- [20] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami. Internet of things (iot): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, 29(7):1645–1660, 2013.
- [21] H. Gulliksson. *Pervasive design*. Håkan Gulliksson, 2015.
- [22] M. Hassenzahl and N. Tractinsky. User experience-a research agenda. *Behaviour & information technology*, 25(2):91–97, 2006.
- [23] S. Hellweger and X. Wang. What is user experience really: towards a ux conceptual framework. *arXiv preprint arXiv:1503.01850*, 2015.
- [24] S. Houde and C. Hill. What do prototypes prototype. *Handbook of human-computer interaction*, 2:367–381, 1997.
- [25] iControl Networks. State of the smart home report, 2015. <http://www.icontrol.com/blog/2015-state-of-the-smart-home-report/>.
- [26] iControl Networks. Ces 2016: Are we getting real about smart home?, January 2016. <https://www.icontrol.com/blog/ces-2016-are-we-getting-real-about-smart-home/>.

- [27] S. Jespersen, R. Stounbjerg, and N. Verdezoto. Ambird: Mediating intimacy for long distance relationships through an ambient awareness system. *Aarhus Series on Human Centered Computing*, 1(1):2, 2015.
- [28] B. Jiang and Y. Fei. Smart home in smart microgrid: A cost-effective energy ecosystem with intelligent hierarchical agents. *Smart Grid, IEEE Transactions on*, 6(1):3–13, 2015.
- [29] P. Jongerius. *Get Agile: Scrum for UX, design & development*. BIS Publishers, 2014.
- [30] D. Kahneman, A. B. Krueger, D. A. Schkade, N. Schwarz, and A. A. Stone. A survey method for characterizing daily life experience: The day reconstruction method. *Science*, 306(5702):1776–1780, 2004.
- [31] R. Khan, S. U. Khan, R. Zaheer, and S. Khan. Future internet: the internet of things architecture, possible applications and key challenges. In *Frontiers of Information Technology (FIT), 2012 10th International Conference on*, pages 257–260. IEEE, 2012.
- [32] J. A. Kientz, S. N. Patel, B. Jones, E. Price, E. D. Mynatt, and G. D. Abowd. The georgia tech aware home. In *CHI’08 extended abstracts on Human factors in computing systems*, pages 3675–3680. ACM, 2008.
- [33] B. Köhler, J. Haladjian, B. Simeonova, and D. Ismailović. Feedback in low vs. high fidelity visuals for game prototypes. In *Proceedings of the Second International Workshop on Games and Software Engineering: Realizing User Engagement with Game Engineering Techniques*, pages 42–47. IEEE Press, 2012.
- [34] T. Kubitza and A. Schmidt. Towards a toolkit for the rapid creation of smart environments. In *End-User Development*, pages 230–235. Springer, 2015.
- [35] M. Kuniavsky. *Smart Things: Ubiquitous Computing User Experience Design: Ubiquitous Computing User Experience Design*. Elsevier, 2010.
- [36] K. Kuusinen. Continuous user experience development. In *INTERACT 2015 Adjunct Proceedings: 15th IFIP TC. 13 International Conference on Human-Computer Interaction 14-18 September 2015, Bamberg, Germany*, volume 22, page 233. University of Bamberg Press, 2015.
- [37] C. Larman and V. R. Basili. Iterative and incremental development: A brief history. *Computer*, 36(6):47–56, 2003.
- [38] E. L.-C. Law, V. Roto, M. Hassenzahl, A. P. Vermeeren, and J. Kort. Understanding, scoping and defining user experience: a survey approach. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 719–728. ACM, 2009.
- [39] N. E. Lee, T. H. Lee, D. H. Seo, and S. Y. Kim. A smart water bottle for new seniors: Internet of things (iot) and health care services. *International Journal of Bio-Science and Bio-Technology*, 7(4):305–314, 2015.

- [40] D. Leonard and J. F. Rayport. Spark innovation through empathic design. *Harvard business review*, 75:102–115, 1997.
- [41] S. Li, L. Da Xu, and S. Zhao. The internet of things: a survey. *Information Systems Frontiers*, 17(2):243–259, 2015.
- [42] J. Liedtka. Perspective: linking design thinking with innovation outcomes through cognitive bias reduction. *Journal of Product Innovation Management*, 32(6):925–938, 2015.
- [43] J. Manyika, M. Chui, J. Bughin, R. Dobbs, P. Bisson, and A. Marrs. *Disruptive technologies: Advances that will transform life, business, and the global economy*, volume 12. McKinsey Global Institute New York, 2013.
- [44] D. Miorandi, S. Sicari, F. De Pellegrini, and I. Chlamtac. Internet of things: Vision, applications and research challenges. *Ad Hoc Networks*, 10(7): 1497–1516, 2012.
- [45] H. Nishikawa, S. Yamamoto, M. Tamai, K. Nishigaki, T. Kitani, N. Shibata, K. Yasumoto, and M. Ito. Ubireal: realistic smartspace simulator for systematic testing. In *UbiComp 2006: Ubiquitous Computing*, pages 459–476. Springer, 2006.
- [46] M. Poppendieck and T. Poppendieck. *Lean software development: an agile toolkit*. Addison-Wesley Professional, ISBN 0-321-15078-3, 2003.
- [47] N. K. Prebensen, S. Rosengren, F. Okumus, and F. Okumus. Experience value as a function of hedonic and utilitarian dominant services. *International Journal of Contemporary Hospitality Management*, 28(1), 2016.
- [48] A. Queirós, A. Silva, J. Alvarelhão, N. P. Rocha, and A. Teixeira. Usability, accessibility and ambient-assisted living: a systematic literature review. *Universal Access in the Information Society*, 14(1):57–66, 2015.
- [49] P. J. Reaidy, A. Gunasekaran, and A. Spalanzani. Bottom-up approach based on internet of things for order fulfillment in a collaborative warehousing environment. *International Journal of Production Economics*, 159: 29–40, 2015.
- [50] C. Rowland, E. Goodman, M. Charlier, A. Light, and A. Lui. *Designing Connected Products: UX for the Consumer Internet of Things*. " O'Reilly Media, Inc.", 2015.
- [51] R. M. Ryan and E. L. Deci. On happiness and human potentials: A review of research on hedonic and eudaimonic well-being. *Annual review of psychology*, 52(1):141–166, 2001.
- [52] M. Schrage. *Serious play: How the world's best companies simulate to innovate*. Harvard Business Press, 2013.
- [53] C. Snyder. *Paper prototyping: The fast and easy way to design and refine user interfaces*. Morgan Kaufmann, 2003.

- [54] L. Tang, Z. Yu, X. Zhou, H. Wang, and C. Becker. Supporting rapid design and evaluation of pervasive applications: Challenges and solutions. *Personal Ubiquitous Comput.*, 15(3):253–269, Mar. 2011. ISSN 1617-4909. doi: 10.1007/s00779-010-0332-6. URL <http://dx.doi.org/10.1007/s00779-010-0332-6>.
- [55] K. Väänänen-Vainio-Mattila, T. Olsson, and J. Häkkinen. Towards deeper understanding of user experience with ubiquitous computing systems: Systematic literature review and design framework. In *Human-Computer Interaction-INTERACT 2015*, pages 384–401. Springer, 2015.
- [56] M. Weiser. The computer for the 21st century. *Scientific american*, 265(3): 94–104, 1991.
- [57] A. Whitmore, A. Agarwal, and L. Da Xu. The internet of things—a survey of topics and trends. *Information Systems Frontiers*, 17(2):261–274, 2015.
- [58] K. Wolf, E. Abdelhady, Y. Abdelrahman, T. Kubitz, and A. Schmidt. mesch: Tools for interactive exhibitions. In *Proceedings of the Conference on Electronic Visualisation and the Arts, EVA '15*, pages 261–269, Swinton, UK, UK, 2015. British Computer Society. doi: 10.14236/ewic/eva2015.28. URL <http://dx.doi.org/10.14236/ewic/eva2015.28>.
- [59] M. Wu, T.-I. Lu, F.-Y. Ling, L. Sun, and H.-Y. Du. Research on the architecture of internet of things. In *Advanced Computer Theory and Engineering (ICACTE), 2010 3rd International Conference on*, volume 5, pages V5–484. IEEE, 2010.
- [60] Z. Yang, Y. Peng, Y. Yue, X. Wang, Y. Yang, and W. Liu. Study and application on the architecture and key technologies for iot. In *Multimedia Technology (ICMT), 2011 International Conference on*, pages 747–751. IEEE, 2011.

Acronyms

Application Protocols - Application Protocols - please consult Al-Fuqaha et al. [1]. 8

Infrastructure Protocol - Infrastructure Protocol - please consult Al-Fuqaha et al. [1]. 8

6LoWPAN - IPv6 over Low power Wireless Personal Area Networks. 8

AAL - Ambient Assisted Living - umfasst Methoden, Konzepte, (elektronische) Systeme, Produkte sowie Dienstleistungen, welche das alltägliche Leben älterer und auch benachteiligter Menschen situationsabhängig und unaufdringlich unterstützen.. 4, 18, 19, 33, 42

Agile - for a definition see chapter 4.1. 45–48, 52, 57, 77

AI - Artificial Intelligence - see chapter 1.1 for a definition.. 4, 42

AmI - Ambient Intelligence - refers to electronic environments that are sensitive and responsive to the presence of people.. 3, 4

API - Application Program Interface - a set of functions and procedures that allow the creation of applications which access the features or data of an operating system, application, or other service.. 61, 67, 68, 71, 72, 74, 75

AR - Augmented Reality - a live direct or indirect view of a physical, real-world environment whose elements are augmented by computer-generated sensory input.. 78, 79

ASR - Automatic Speech Recognition - (ASR) can be defined as the independent, computer-driven transcription of spoken language into readable text in real time (Stuckless, 1994).. 61, 68

B2B - business to business - refers to a situation where one business makes a commercial transaction with another.. 57

Big Data - see chapter 1.1 for a definition.. 4

BLE - Bluetooth Low Energy - is a wireless personal area network technology.. 6, 8, 9, 41, 69, 72, 74, 75

CDR - Central Design Record central artefact that documents design agreements for the whole design and developer team.. 53

CIP - Continuous Interdisciplinary Prototyping - for further information please see chapter 4.2. 53, 55, 58, 61, 62, 64, 66, 68

Cloud Computing - see chapter 1.1 for a definition.. 4

cloud service - an application that is executed on a remote server and in most cases accessible via an API. 11

CoAP - Constrained Application Protocol. 9

Continous Development - see Kuusinen [36]. 52, 62

COP - Common Operating Picture - for further information please see chapter 3. 31

EPC - Electronic Product Code - is designed as a universal identifier that provides a unique identity for every physical object anywhere in the world, for all time.. 8

EXI - Efficient XML Interchange format. 9

ExP - Experience Prototyping - for information see chapter 2.3. LXXXIII, 2, 20, 21, 23, 25, 48, 78

functional ExP - functional Experience Prototyping/functional Experience Prototype - for our definition please see chapter 2.6. LXXXII-LXXXIV, 2, 26, 27, 41, 45, 51-65, 71, 73, 74, 76, 77, 79

GUI - Grapical User Interface - a visual way of interacting with a computer.. 21-23, 25, 40, 41, 43, 65, 66, 68, 73, 74

HCI - Human Computer Interaction - researches the design and use of computer technology, focusing on the interfaces between people (users) and computers.. 18, 62

HTTP - Hypertext Transfer Protocol. 9

I/O - Input/Output abbreviation for input and output devices and interfaces.. 60, 65, 67, 70

IDE - Integrated Development Environment. 9, 13

IEEE 1905.1 - IEEE 1905.1 is a communication platform developed by Google. 9

ILE - Intelligent Living Environments - a subset of Smart Environments defined within this work in chapter 1.1. LXXXII-LXXXIV, 2-4, 6, 15, 20-27, 29-38, 40-42, 45, 51-55, 57, 59-61, 63, 68, 70, 71, 73, 74, 76-78

IM bot - Instant Messaging Bots - IM bot is a chatterbot program that uses instant messaging as an application interface.. 68

- IoT** - Internet of Things - the network of physical devices, vehicles, buildings and other items—embedded with electronics, software, sensors, and network connectivity that enables these objects to collect and exchange data.. LXXXIII, LXXXIV, 1, 4, 6, 7, 9, 11, 13, 15, 17, 22, 35, 56, 71, 72, 74, 77, 78
- IP** - Internet Protocol - communications protocol for computers connected to a network, especially the Internet, specifying the format for addresses and units of transmitted data.. 74
- IPA** - Intelligent Personal Assistant - a software agent that can perform tasks or services for an individual.. 38, 40–42
- IPv4** - Internet Protocol version 4 is the fourth revision of the Internet Protocol and is the most common version used today.. 8
- IPv6** - Internet Protocol version 6 is the most recent version of the Internet Protocol. 8
- IT** - Information Technology. 1, 6, 11, 24, 79
- IxD** - Interaction Design - s defined as "the practice of designing interactive digital products, environments, systems, and services.". LXXXIII, 23, 32, 35, 55, 57, 77
- JSON** - JavaScript Object Notation. 9
- Kanban** - for more information see chapter 4.1. 48
- KUI** - Kinetic User Interface - an emerging type of user interfaces that allow users to interact with computing devices through the motion of objects and bodies.. 40
- Lean** - for a definition see chapter 4.1. 45, 47, 48, 52, 53, 62, 77
- LiFi** - Light Fidelity - is a bidirectional, high-speed and fully networked wireless communication technology similar to Wi-Fi.. 8
- LoRaWAN** - (Long Range Wide Area Network) is a network protocol by the LoRa Alliance. 8, 9
- micro-controller** - A microcontroller (or MCU, short for microcontroller unit) is a small computer (SoC) on a single integrated circuit containing a processor core, memory, and programmable input/output peripherals.. 9
- middleware** - software that acts as a bridge between an operating system or database and applications, especially on a network.. 11
- mixed Reality** - mixed Reality - is the merging of real and virtual worlds to produce new environments and visualizations where physical and digital objects co-exist and interact in real time.. 78, 79

MQTT - Message Queue Telemetry Transport. 9, 72, 74, 75

Neural Networks - a family of models in Machine Learning inspired by biological neural networks.. 75

NFC - Near Field Communication - is a set of communication protocols that enable two electronic devices, one of which is usually a portable device, to establish communication by bringing them within about 4 cm (2 in) of each other.. 6, 8, 69

open source - source code made available with a license in which the copyright holder provides the rights to study, change, and distribute the software to anyone and for any purpose.. 71, 74

OS - operating systems is the collection of software that directs a computer's operations, controlling and scheduling the execution of other programs, and managing storage, input/output, and communication resources.. 11

OWL - Web Ontology Language is a semantic web standard by W3C. 9, 74

Pair Programming - Pair programming is an agile software development technique in which two programmers work together at one workstation. 45

Physcial Computing - see chapter 1.1 for a definition.. 3, 4

RDF - Resource Description Framework. 9

REST - REpresentational State Transfer. 9

RFID - Radio Frequency Identification - uses electromagnetic fields to automatically identify and track tags attached to objects.. 6, 8

Scrum - for a definition see chapter 4.1. 45, 47, 48, 52, 57

SDK - Software Development Kit - typically a set of software development tools that allows the creation of applications for a certain development platform.. 68

single-board computer - A single-board computer (SBC) is a complete computer built on a single circuit board, with microprocessor(s), memory, input/output (I/O) and other features required of a functional computer.. 9, 11

Smart Environment - evolves from the definition of Ubiquitous Computing. LXXXII, 2-6, 22, 71, 77, 79

home automation - home automation - automation of devices, chores and activities at home.. 1

Smart Home - Smart Home - currently popular term for for systems.. 1, 4, 15, 17, 42, 43

Smart Home hub - Smart Home hub - Hub connected to the internet and other smart home appliances.. [2](#), [15](#)

speech synthesis - the process of generating spoken language by machine on the basis of written input.. [61](#), [68](#), [75](#)

TCP - Transmission Control Protocol. [9](#)

Technical Protoyping - for our definition see chapter [2.1](#). [13](#), [24](#), [27](#)

TUI - Tangible User Interface - a user interface in which a person interacts with digital information through the physical environment.. [40](#)

UbiComp - Ubiquitous Computing - see chapter [1.1](#) for a definition.. [3](#), [4](#), [6](#), [17-19](#), [22](#), [26](#), [49](#), [54](#), [71](#)

UCD - User-centered Design - a framework of processes in which the needs, wants, and limitations of end users of a product, service or process are given extensive attention at each stage of the design process. [45](#), [56](#), [77](#)

UDP - User Datagram Protocol. [9](#), [72](#)

UI - User Interface - the means by which the user and a computer system interact.. [54](#), [72](#)

URI - Unique Resource Identifier. [9](#)

usability - usability - Usability is the ease of use and learnability of a human-made object. In software engineering, usability is the degree to which a software can be used by specified consumers to achieve quantified objectives with effectiveness, efficiency, and satisfaction in a quantified context of use.. [18](#)

UX - User Experience - various definitions available such as: All the aspects of how people use an interactive product: the way it feels in their hands, how well they understand how it works, how they feel about it while they're using it, how well it serves their purposes, and how well it fits into the entire context in which they are using it. - Alben [2]. [LXXXII-LXXXIV](#), [2-4](#), [6](#), [17-19](#), [22-27](#), [29-33](#), [35-37](#), [40](#), [41](#), [45](#), [48-50](#), [52-54](#), [56-59](#), [62](#), [77](#), [79](#)

VR - Virtual Reality - is a computer technology that replicates an environment, real or imagined, and simulates a user's physical presence and environment to allow for user interaction.. [78](#)

Weave - Weave is a communication platform developed by [Google](#). [9](#), [74](#)

WiFi - WiFi is standard for WLAN but commonly used especially in the US as synonym for WLAN. [6](#), [8](#), [9](#), [72](#)

WSN - Wireless Sensor Networks - are spatially distributed autonomous sensors to monitor physical or environmental conditions.. [6](#)

XML - Extensible Markup Language. [9](#)

Appendix A

List of Identifiers

[CO.1 - 5]	List of co ntextual dimensions for ILEs as part of the UX dimension "Context of Use" → see p. 35
[Comp.1 - 15]	Tool co mponents for a functional ExP tool kit → see p. 65
[CR.1 - 8]	Ch aracteristics of IoT → see figure 2.1 on p. 7
[DesMeth]	The optimized de sign and development me thod for functional ExP defined in this work. → see chapter 4.2
[Eval.1 - 7]	Different common user eva luation methods within research and industry → see p. 50
[EX.1 - 37]	Various IoT application ex amples within research and industry → see figures 2.9 and 2.9 on p. 16 and 17
[ExP.1 - 12]	Different common ExP methods within research and industry → see figure 2.4 for details on p. 21 for details
[I.1 - 5]	List of general I ntentions of Use for ILEs as part of the UX dimension "Intentions of Use" → see p. 33
[IxD.1 - 5]	List of IxD aspects for ILEs as part of the UX dimension "System Characteristics" → see p. 37
[IxD.Ex.1 - 4]	List of IxD aspects for ex plicit interaction within ILEs as part of the UX dimension "System Characteristics" → see p. 37ff
[IxD.Im.1 - 4]	List of IxD aspects for im plicit interaction within ILEs as part of the UX dimension "System Characteristics" → see p. 37ff
[MainObj]	Main o bjective of this work → see p. 26
[OptiM.1 - 5]	Aspects of an design and development opti mized m ethod for incorporating functional ExP → see p. 53

- [Q.1 - 3] Research and design **q**uestions that need to be answered to pursue [MainObj] of this work
→ see p. 26
- [Spec.1 - 7] **S**pecification for a functional ExP tool kit
→ see p. 61
- [TCL.1 - 7] **T**echnical **c**hallenges of IoT
→ see figure 2.1 on p. 7
- [UC.1 - 6] List of use case examples for ILEs
→ see p. 42
- [UX.1 - 6] List of **UX** Design aspects for ILEs as part of the UX dimension "System Characteristics"
→ see p. 36
- [UxOut] The **UX outline** for ILEs defined in this work.
→ see chapter 3.1

Appendix B

Evaluation Documentation

- Interview Structure:
 - Background of executing researcher
 - Background of interviewee
 - Introduction to graduation topic
 - Introduction to concept of functional Experience Prototyping
 - Question: Do you have any question on the concept of functional Experience Prototyping?
 - Bridge to design & development process using fExP
 - Question: What is the typical design & development process used by expert/researcher?
 - Introduction to design & development process using fExP
 - * agile
 - * user-centered design - focus on added value for user - fulfillment of a user need or solving a problem
 - * conception & creation phase
 - * conception
 - iterative development of concept
 - process can jump back to any step caused by new insights through design workshops or user tests
 - first enters the creation phase is concept is validated
 - * creation
 - continuous development - no fixed time frame for sprints
 - sprints of ux tasks and technical development can be asynchronous
 - * prototype
 - Starts with low fidelity experience prototyping and technical feasibility studies in the conception phase
 - A scenarios based functional experience prototype is created in the end of the conception phase for concept validation
 - This scenario fExP is extended provide the UX of the full concept
 - Question: What is your opinion on this process?

- Question: Do you think it is realistic and applicable in a real world setting?
- Question: What points or aspects would you identify as critical or challenging (in a real world setting)?
- Question: Do you have any other feedback or remarks on the design & development process using fExP?
- Question: Do you have any other feedback or remarks on this graduation project?
- Question: Any other question or topic you would like to discuss?
- Interview/focus group discussion will be recorded preferably as video recording and at least as an audio recording.
- Answers and other important elements of the interview will be documented in a transcript

