Niko Mehtonen

INTERNET OF THINGS SERVICE DEVELOPMENT EXPECTATIONS

JYVÄSKYLÄN YLIOPISTO
TIETOJENKÄSITTELYTIETEIDEN LAITOS
2018

TIIVISTELMÄ

Mehtonen, Niko
Esineiden Internetin palveluiden kehittämisen odotukset
Jyväskylä: Jyväskylän yliopisto, 2018, 56 s.
Tietojärjestelmätiede, pro gradu -tutkielma
Ohjaaja: Rousi, Rebekah

Esineiden Internet on visio tulevaisuudesta, jossa jokapäiväiset esineet levittäytyvät Internettiin, kommunikoiden ja tehden yhteistyötä toistensa kanssa saavuttaakseen yhteisiä tavoitteita. Esineiden Internetin palveluita kehittävät ohjelmoijat käyttävät ohjelmistokehyksiä ja työkaluja osana kehitystyötään. Heidänkin työhönsä vaikuttaa käyttäjäkokemus. Käyttäjäkokemus on monitulkintainen käsite, joka on noussut vastaliikkeenä tehtävä- ja työpainotteiselle käytettävyyden käsitteelle. Odotukset ovat avainasemassa käyttäjäkokemuksen arvioimisessa.

Tässä tutkielmassa esineiden Internet-palvelua kehittävän ryhmän odotuksia ja kokemuksia kerättiin kyselyn avulla. Tutkimuskysymyksenä oli: ”Kuinka esineiden Internetin palveluiden kehittämisen käyttäjäkokemuksen odotukset eroavat niiden kehittämisen todellisuudesta?” Lisäkysymyksenä oli: ”Kuinka nämä odotukset vaikuttavat käyttäjäkokemukseen?”

Tämän Pro Gradu-tutkielman tulokset antavat ymmärtää, että ohjelmoijan odotettujen vahvuuksien kohtaamatta jättämisellä vaikutti olevan suurempi vaikutus odotusten arvioimisen tuloksiin kuin odotettujen ongelmien kohtaamatta jättämisellä. Tämän perusteella vaikuttaisi siltä, että jos odotettujen vahvuuksien kohtaamisessa on puutteita, on sillä suurempi vaikutus ohjelmoijan käyttäjäkokemukseen kuin jos odotetut ongelmat jäävät kohtaamatta. Suunnittelun kehityksen prosessien kannalta vaikuttaisi siltä, että olisi parempi valmistaa ohjelmoijat vastoinkäymisiin ja vähemmän sensaatiomaisiin tuloksiin, kuin että innostaisi odottamaan paljon ennen kuin kehitysprosessi on toteutettu.

Asiasanat: esineiden internet, käyttäjäkokemus, odotukset, kehittäjät käyttäjinä

ABSTRACT

Mehtonen, Niko
Internet of Things service development expectations
Jyväskylä: University of Jyväskylä, 2018, 56 p.
Information Systems, Master's Thesis
Supervisor: Rousi, Rebekah

Internet of Things (IoT) is a vision of the future where everyday objects are extending to the Internet, communicating and cooperating with each other to achieve common goals. Internet of Things is gaining ground as a novel paradigm in modern wireless telecommunications. Programmers developing IoT services use the frameworks and tools in their development work to create these services. As such, their work is impacted by user experience (UX). User experience as a term has a broad range of meanings and it has risen as a counter-movement to the task and work orientated usability. Expectations play a key role in user experience evaluation.

In this thesis, data was collected via questionnaire format regarding the experiences and expectations of a group of programmers developing an IoT service. The research question, from a programmer's perspective, was: "How do the expectations for user experience of Internet of Things service development differ to the actual reality of developing these services? "A secondary research question was: "How do these expectations affect the user experience?"

The results of this Masters' research suggest that there is a stronger impact on the programmers' experience of service development when the expectations of strengths are not met, as compared to when expectations of problems are not met. This indicates that if there are shortcomings in meeting positive expectations the impact on programmer user experience is greater than if negative expectations are not met. For the design and development process this means that it is perhaps better to gear programmers towards being prepared for setbacks or less sensational outcomes, than it is to instill highly positive expectations before a development process is fulfilled.

Keywords: internet of things, user experience, expectations, programmers as users, user programmers

TABLES

CONTENTS

1 INTRODUCTION

Internet of Things (IoT) is a vision, or a paradigm, of the future where everyday objects are extending to the Internet by a wireless network (Welbourne et al., 2009). These objects are able to communicate and cooperate with each other to achieve common goals (Atzori, Iera, & Morabito, 2010). The novelty of IoT is not necessarily in the objects themselves, but in the expected amount of them: from billions to even trillions of connected smart objects (Kopetz, 2011). Internet of Things is gaining ground as a novel paradigm in modern wireless telecommunications (Atzori et al., 2010).

IoT has numerous application possibilities, with greatest market shares projected to be in health care and manufacturing (Al-Fuqaha, Guizani, Mohammadi, Aledhari, & Ayyash, 2015). In health care, example applications for IoT monitoring and managing medical equipment, managing medical information and monitoring vital signs (Hu, Xie, & Shen, 2013). IoT in manufacturing enables machines to monitor themselves and by collecting and analyzing production data and causes of issues (Al-Fuqaha et al., 2015).

Examples for everyday life include smart refrigerators that follow expiration dates of food items and places an order if the food item is running low (Kopetz, 2011) or clothes dryer completing its cycle just in time to provide clothes for the work day (Hurlburt, Voas, & Miller, 2012).

User experience (UX) as a term is associated with an array of meanings (Forlizzi & Battarbee, 2004) with assorted definitions and perspectives approaching the concept from diverse points of view (Roto, Law, Vermeeren, & Hoonhout, 2011). User experience has gained ground mostly as a countermovement to the task-focused paradigm of usability (Hassenzahl & Tractinsky, 2006). While traditional human computer interaction in the form of usability is focused on designing to reduce pain, user experience tries to improve our lives by designing pleasure (Hassenzahl & Tractinsky, 2006). User experience should be evaluated before and during interaction, not just after (Vermeeren et al., 2010). However, user experience does not have one common, exact definition, which impairs understanding the concept (Forlizzi & Battarbee, 2004; Ibargoyen, Szostak, & Bojic, 2013; Law, Roto, Hassenzahl, Vermeeren, & Kort, 2009; Väänänen-Vainio-Mattila, Roto, & Hassenzahl, 2008).

Programmers developing IoT services are users themselves. They use the frameworks and tools in their development work to create theses services. As such, their work is impacted by user experience. However, this seems to be limited research from this point of view. Expectations play a key role in user experience evaluation (Roto, 2007). For this reason, the research question for this thesis is: "How do the expectations for Internet of Things service development user experience differ from the actual reality of developing them?" A secondary research question is: "How do these expectations affect the user experience?"

The structure of this paper is as follows: Chapter 2 opens us the concept of IoT, followed by explanations of software architecture and design examples of IoT. After this, challenges and limitations are collected and then energy efficiency and security are explored. In chapter 3, user experience is explained, starting with a brief look into usability, followed by an explanation of user experience and some definitions for it. These are followed by hedonism and pragmatism in user experience, components of user experience, anticipation and expectations in relation to user experience, how to evaluate user experience, some critique and limitations of user experience and a look into programmers as users. Chapter 4 explores how to design user experience for Internet of Things and design principles and guidelines for IoT systems. Chapter 5 outlines research methods: how the questionnaire was developed, how participants were recruited and ethical considerations related to the empirical research. Chapter 6 provides the results and discussion on them alongside limitations. Chapter 7 presents the conclusion and possible future research topics.

2 INTERNET OF THINGS

In this chapter, Internet of Things is explored. First, IoT is explored on a conceptual level, followed by a look into software architecture and design for IoT. After that is a collection of greatest challenges and limitations facing IoT followed by sections on energy efficiency and security of IoT.

2.1 Understanding Internet of Things

In the early stages Internet of Things, a term coined by Kevin Ashton in 1999, referred to Radio Frequency Identification (RFID) technology (Ashton, 2009). It was later associated with other technologies such as sensors, actuators and GPS devices (Da Xu, He, & Li, 2014).

Internet of Things changes the dimensions of communication from anytime, anyplace and *anyone* to anytime, anyplace and *anything* (Coetzee & Eksteen, 2011; Tan & Wang, 2010). Communication is now a mixture of person-to-person, person-to-machine and machine-to-machine (M2M) communication (Al-Fuqaha et al., 2015). M2M, alongside then Internet and mobile technologies can be seen as the beginning for IoT (Al-Fuqaha et al., 2015). In Internet of Things, the word "Thing" refers to the information of the thing while "Internet" refers to an Internet application (Huang & Li, 2010).

Internet of Things has been called both a vision of future (e.g. Kopetz, 2011; Kortuem, Kawsar, Fitton, & Sundramoorthy, 2010; Miorandi, Sicari, De Pellegrini, & Chlamtac, 2012) and a paradigm (e.g. Atzori et al., 2010; Chen, 2012; Salman, Abu-Issa, Tumar, & Hassouneh, 2015). However, it can be both. Atzori et al. (2010) see IoT as a paradigm consisting of three visions with different orientations: the "Things"-oriented vision, "Internet"-oriented vision and "Semantic"-oriented vision. In "Things"-oriented vision, concentration is on the everyday objects being introduced to a common framework, while in "Internet"-oriented vision focuses on networks (Atzori et al., 2010). Whereas the "Semantics"-oriented vision centers on addressing each object uniquely and representing and storing of the exchanged information as challenging issues (Atzori et al., 2010).

2.2 Software architecture and design

There are a variety of different IoT software architectures for networking objects consisting of different layers. Al-Fuqaha et al. (2015) identify four types: three-layer, middleware based, service-oriented architecture (SOA) based and five-layer. All of these architectures have commonalities, such as an application layer of some sort.

Three-layer architecture is a basic model (Al-Fuqaha et al., 2015) consisting of Application, Network and Perception layers (Yang et al., 2011). Perception layer is for collecting information, while network layer transports this information over the Internet (Yang et al., 2011). Application layer is the topmost layer and it is for discovering and taking on services (Yang et al., 2011).

Middleware based architecture is meant to be an abstract layer hiding technological details such that application developers can focus on developing the applications (Chaqfeh & Mohamed, 2012). It simplifies the development of IoT services and integrating legacy technologies with new ones (Atzori et al., 2010). Architecture proposed by Tan and Wang (2010), the middleware based architecture used as an example in Al-Fuqaha et al. (2015), incorporates a coordination layer below the middleware layer to account for the lack of global standards for communication between different application systems. Even if global standards are created for communication, a coordination layer is required for facilitating communications between new standard-following applications and old non-standard applications (Tan & Wang, 2010). These middleware based architectures have similarities with the SOA based ones (Atzori et al., 2010).

Service-oriented architecture allows for the separation of complex systems into an ecosystem of applications with simpler, clearer components (Atzori et al., 2010). The layers are application, service composition, service management, object abstraction and objects (Atzori et al., 2010). For SOA, existing standards should not be used as they are because they have been designed for a different scenario (Guinard, Trifa, Karnouskos, Spiess, & Savio, 2010). Instead, SOA standards and tools need to be simplified, optimized and adapted for IoT needs (Guinard et al., 2010). Benefits of SOA are reduced need of gateways and translation between components (Guinard et al., 2010). A challenge for developers is in "service discovery", finding adequate services to complete a given task (Guinard et al., 2010). Because SOA does not place restrictions on a specific technology, it also enables reuse of software and hardware (Atzori et al., 2010).

While other architectures have had application layer on top, five-layer architecture has business layer topmost, followed by application layer. (Al-Fuqaha et al., 2015) Below these are service management, object abstraction layer and objects layer (Al-Fuqaha et al., 2015). Object layer, also called perception layer like in three-layer architecture, are the physical sensors collecting and processing data for the IoT while object abstraction layer, also called network or transmission layer, represents various technologies used for transferring information (Al-Fuqaha et al., 2015; Khan, Khan, Zaheer, & Khan, 2012). Service management layer is also called middleware layer and it offers the services, while application layer provides the requested services based on the information processed in the previous layer (Al-Fuqaha et al., 2015; Khan et al., 2012). Business layer is for managing the IoT services overall and it enables building of graphs, business models and Big Data analysis for decision-making support (Al-Fuqaha et al., 2015; Khan et al., 2012).

2.3 Challenges and limitations

There are several challenges specific to the requirements of IoT (Matharu, Upadhyay, & Chaudhary, 2014). These challenges vary from policy and other contextual challenges to applications and technical challenges (Coetzee & Eksteen, 2011). Taking a small sample of ten articles discussing IoT challenges, clear trends can be seen. TABLE 1 displays the most common challenges (ones appearing in at least two articles), their occurrences and where they occurred. Articles used for this comparison were the ones listing challenges. Many of the challenges were joined based on pairings done in original articles, e.g. standardization and interoperability. Others were joined based on common topic, while the articles referred to them with different terms, e.g. scale, Big Data and data deluge.

TABLE 1: Challenges of IoT

Challenge	Number of occurrences	References
Standardization or interoperability	8	(Atzori et al., 2010; Bandyopadhyay & Sen, 2011; Coetzee & Eksteen, 2011; Elkhodr, Shahrestani, & Cheung, 2013; Hurlburt et al., 2012; Khan et al., 2012; Matharu et al., 2014; Miraz, Ali, Excell, & Picking, 2015)
Privacy or identity management	8	(Atzori et al., 2010; Bandyopadhyay & Sen, 2011; Coetzee & Eksteen, 2011; Elkhodr et al., 2013; Hurlburt et al., 2012; Khan et al., 2012; Matharu et al., 2014; Saeed, Ammar, Harras, & Zegura, 2015)
Data integrity or security	6	(Atzori et al., 2010; Bandyopadhyay & Sen, 2011; Coetzee & Eksteen, 2011; Hurlburt et al., 2012; Khan et al., 2012;

		Matharu et al., 2014)
Naming	4	(Atzori et al., 2010; Elkhodr et al., 2013; Khan et al., 2012; Matharu et al., 2014)
Scale, Big Data or data deluge	3	(Coetzee & Eksteen, 2011; Hachem, Teixeira, & Issarny, 2011; Matharu et al., 2014)
Costs	2	(Hurlburt et al., 2012; Saeed et al., 2015)
Authentication or access control	2	(Atzori et al., 2010; Coetzee & Eksteen, 2011)
Energy efficiency or greening	2	(Khan et al., 2012; Miraz et al., 2015)

Standardization and interoperability are one of the two most common challenge areas, appearing in all but two of the articles, as seen in TABLE 1. Standardization is needed for the interoperability of IoT devices and services (Coetzee & Eksteen, 2011). There are several standardization efforts (Al-Fuqaha et al., 2015). Different standardization bodies, alongside other interested parties, approach IoT from either an "Internet oriented" or "Things oriented" perspective, resulting into differing visions (Atzori et al., 2010). Some of these standardization bodies include Internet Engineering Task Force (IETF) World Wide Web Consortium (W3C), EPCglobal, Institute of Electrical and Electronics Engineering (IEEE), International Organization for Standardization (ISO) and European Telecommunications Standards Institute (ETSI) (Al-Fuqaha et al., 2015; Atzori et al., 2010).

Other most common challenge area was privacy and identity management, appearing in all but two articles as well in TABLE 1. Identity management can refer to either identity of users (Elkhodr et al., 2013) or objects (Kanuparthi, Karri, & Addepalli, 2013; Khan et al., 2012; Matharu et al., 2014). The latter meaning is used in this thesis. Privacy is a central issue to IoT, because more details about people are being collected with the increasing number of connected devices (Saeed et al., 2015).

Other challenges mentioned were security and data integrity with six mentions in TABLE 1. Security was often paired with privacy issues. Chapter 2.5 below further discusses security and privacy.

Naming had four mentions, as seen in TABLE 1. The purpose of naming traditionally is to translate IP addresses into human-readable names and back (Elkhodr et al., 2013). However, with IoT, objects are often communicating with each other (Atzori et al., 2010).

Scale, Big Data and data deluge are comparable topics, totaling three mentions. They all involve large amounts of data to be handled in different ways. Costs had two mentions and authentication alongside access control had two mentions as well. Sensor energy usage and greening of IoT had two mentions (Khan et al., 2012; Miraz et al., 2015). In addition to these, there were a handful of other challenges mentioned only once. They are not listed here to keep focus on the biggest issues.

2.4 Energy efficiency

By 2020 there is estimated to be over 50 billion devices connected to the Internet (Evans, 2011). Most of these will be battery powered (Jayakumar et al., 2014). There are several areas of interest in increasing energy efficiency, such as ultra-low power hardware platforms, intelligent system-level power management techniques (Jayakumar et al., 2014) and efficient wireless communication stacks (Palattella et al., 2013). Using peer-to-peer traffic over a centralized model can also be used to reduce the power cost for useful traffic (Kardeby, Jennehag, & Gidlund, 2015). For hardware, because most of the time battery-powered IoT devices are in sleep, it is important to use components which have very low power consumption in both sleep mode and during active computations (Jayakumar et al., 2014).

There are several alternatives for wireless communication standards, such as Wi-Fi, Bluetooth Low Energy (BLE), ZigBee and 6LoWPAN (Jayakumar et al., 2014; Siekkinen, Hiienkari, Nurminen, & Nieminen, 2012). The last two are built on the IEEE 802.15.4 standard (Jayakumar et al., 2014). Both BLE and ZigBee consume very little energy (Siekkinen et al., 2012). Jayakumar et al. (2014) foresee BLE being likely the wireless standard of choice for IoT devices. Bluetooth is already ubiquitous through smartphones, so this would make sense. Matharu et al. (2014) see IPv6 being the protocol of choice. These two predictions are not necessarily exclusionary (Nieminen et al., 2015).

Even with all advances in energy efficiency, it would be unreasonable to attempt changing batteries for all 50 billion devices (Chen, 2012; Miraz et al., 2015). Research into minimizing energy consumption has been done extensively, but that alone is insufficient as finding ways to harvest energy from the environment is needed (Meng & Jin, 2011). Energy harvesting can significantly prolong the lifetime of a system, possibly making some of them self-sufficient (Jayakumar et al., 2014).

One possibility of powering low-powered communication devices is ambient backscattering, presented by Liu et al. (2013). They use ambient radio frequency signals, such as TV broadcast or cellular transmission, to communicate between two battery-free devices (Liu et al., 2013). This is similar to RFID tags, but does not require an RFID reader as a specialized power source (Liu et al., 2013). These kinds of devices have been shown to be capable of communicating over distances of tens of meters and through multiple walls (Parks, Liu, Gollakota, & Smith, 2014). And this might be only the beginning. However, there are tradeoffs to this technology, e.g. that longer range communications require the other endpoint to be powered (Gollakota, Reynolds, Smith, & Wetherall, 2014). Also, the act of backscattering reduces the available power to be harvested (Gollakota et al., 2014).

2.5 Security

There is a broad spectrum of targets for attacking in IoT (Covington & Carskadden, 2013). Gamundani (2015) recognize four types of attacks on IoT: application based, connection based, platform based and other forms of attack. With many IoT devices being constrained in storage and other capabilities, thus lacking comprehensive security software, application based attacks exploit this flaw (Gamundani, 2015). Connection based attacks take advantage of the unsecure nature of the Internet combined to the lack of security measures in IoT devices (Gamundani, 2015). This lack of security can compromise information about other hidden devices which are somehow connected to the unsecure IoT device broadcasting information elsewhere (Gamundani, 2015). Third type of attack, platform based, is made possible by some platforms having security issues by their very nature (Gamundani, 2015). These platforms can be for example cloud computing or communications platforms. Last category exists, because the first three categories may not be exhaustive (Gamundani, 2015). These categories can also be combined in a single attack for more devastating results (Gamundani, 2015).

Cyber-attacks have implications, of which Covington & Carskadden (2013) categorize three varieties related to IoT. The first category is capture, second category is collectively referred to as disrupt attacks while the third category is manipulate. Capture can refer to either capturing and accessing information or capturing and gaining control of physical or logical systems, to give the attacker an advantage over the target (Covington & Carskadden, 2013). Gaining control of the system can lead into gaining access to information (Covington & Carskadden, 2013). Connection based attacks would be suitable for gaining access to information while platform based attacks could be used for gaining control of systems. Disrupt attacks, also called degrade, deny or destroy attacks, are used to induce disadvantage to the target (Covington & Carskadden, 2013). Capturing a system can also offer the attackers an opportunity to disrupt in (Covington & Carskadden, 2013). Application based attacks are opportune here, being able to cause interference on data read, resulting in decisions being founded on false information (Gamundani, 2015). Manipulate attacks are intended to influence the decision made by the target (Covington & Carskadden, 2013). This can be done by either influencing outside information, by manipulating sensor gathering the information or by interceding the communications between entities (Covington & Carskadden, 2013). Application based attacks are opportune here again, based on the same reason of causing decisions to be made on false or biased information (Gamundani, 2015), alongside previously mentioned communications based.

With the swift advancements in IoT, construction of security in the IoT architecture is a topic of utmost importance (Matharu et al., 2014). While security of IoT is being researched, there are various security aspects which still require deeper scrutiny (Matharu et al., 2014).

3 USER EXPERIENCE

In this chapter, user experience is defined both as a concept and as a field of research. First, usability is described as background, followed by an explanation of user experience as a concept, its definition and its relation to usability, hedonism and pragmatism in user experience, components of user experience, anticipation and expectations, how user experience can be evaluated and a short section on critique and limitations of user experience. The last section takes a look into programmers as users.

3.1 Usability

Usability in the development of systems has been researched for quite some time. Wallach and Scholz (2012) see the paper "Designing for Usability: Key principles and What Designers Think" by Gould and Lewis (1985) as a seminal piece of work, founding the principles of user-centered design. Gould and Lewis (1985) recommend three principles for designing usable systems: early and continuous focus on users, usage of empirical measurements and iterative design. Here iterative design means that the system should be modified and tested alternatively over and over again so that the problems can be fixed (Gould & Lewis, 1985). The researchers started recommending these principles already during the 1970's, so the topic was not entirely new during the 1980's (Gould & Lewis, 1985).

Nielsen (1993, pp. 26) defines five attributes for usability: learnability, efficiency, memorability, errors and satisfaction. Learnability means that the system should be easy to learn so that the user can start working with the system rapidly (Nielsen, 1993). Efficiency means that once learned, the system should enable high levels of productivity (Nielsen, 1993). Memorability means that a casual user can return to using the system after a period of not using it and not have to relearn everything (Nielsen, 1993). Errors mean that the system should have a low rate of errors and the errors should be easily recovered from (Nielsen, 1993). Additionally, there must not occur any catastrophic errors (Nielsen, 1993). Satisfaction means that the system should be pleasurable to use, subjectively satisfying the users when they are using it (Nielsen, 1993). The last attribute, satisfaction, is very important in user experience. Usability and user experience are intertwined to each other (Vermeeren et al., 2010).

3.2 User Experience

User experience and usability have fundamental differences (Bevan, 2009). User experience has gained momentum as a countermovement to the more task and work centered usability paradigm (Hassenzahl & Tractinsky, 2006). Instead, user experience focuses on improving the quality of life by designing pleasure, instead of absence of pain which is more common with usability (Hassenzahl & Tractinsky, 2006). This is realized by focusing on the user affect, sensation, meaning and value of interactions with technology (Law et al., 2009). However, user experience has no commonly agreed upon exact definition, which impairs on understanding the concept (Ibargoyen et al., 2013; Law et al., 2009; Väänänen-Vainio-Mattila et al., 2008).

Usability as a framework is limited: it focuses mainly on the user cognition and user performance (Law et al., 2009). In contrast, user experience highlights the non-utilitarian aspects such as user affect, sensations and the importance and value of such interactions in the daily life (Law et al., 2009). User experience consists of different time spans and can vary from anticipated to cumulative (Roto et al., 2011).

3.3 Definitions for user experience

For this thesis, four different definitions for user experience have been chosen to be compared and contrasted. These definitions can be seen in TABLE 2 below.

TABLE 2: Definitions for user experience

Source	Definition
(Hassenzahl & Tractinsky, 2006)	"a consequence of a user's internal state […], the characteristics of the designed system […] and the context (or the environment) within which the interaction occurs"
(Hassenzahl, 2008)	"a momentary, primarily evaluative feeling (good-bad) while interacting with a product or service"
ISO-standard 9241-210:2010 (International Organization	"person's perceptions and responses resulting from the use and/or anticipated use of a

for Standardization, 2010)	product, system or service"
(Roto et al., 2011)	"the experience(s) derived from encountering systems"
(Fronemann & Peissner, 2014)	"an evaluative feeling of users interacting with a product or service"

All the preceding definitions are connected by user and their reaction when using a system, service or product. The second definition (Hassenzahl, 2008) does not use the term user, but someone must experience the feeling mentioned in the definition. Second (Hassenzahl, 2008) and fifth (Fronemann & Peissner, 2014) definitions both strongly highlight an evaluative feeling. Fourth definition (Roto et al., 2011) is simple, but provides further clarification afterwards. Encountering refers to "using, interacting with, or being confronted passively" (Roto et al., 2011). By their definition, user experience does not require one to actively use or interact with the system. The first definition (Hassenzahl & Tractinsky, 2006) focuses on the consequences, which can also be seen in the ISO-standard (2010) as responses. Nonetheless, perceptions are also involved, which are comparable to an evaluative feeling. Perceptions are not based on feelings, so they can also lean toward usability. Therefore, in addition to common elements, there are significant differences and differing approaches.

3.4 Hedonism and pragmatism in user experience

As a subjective source of pleasure, user experience contains many hedonic qualities (Diefenbach & Hassenzahl, 2011; Hassenzahl, Diefenbach, & Göritz, 2010). Hedonic qualities refer to judging the quality of a product based on how well it can potentially support pleasure in use and ownership, fulfilling what can be called "be-goals" (Hassenzahl et al., 2010). Counter to this, pragmatic qualities refer to judging the quality of a product based on how well it fulfills so-called "do-goals" (Hassenzahl et al., 2010). Pragmatic qualities can be compared the four first attributes for usability by Nielsen (1993) through the lens of the fifth attribute, satisfaction. Pragmatic qualities are compared to usability by Hassenzahl et al. (2010) as well. Hedonic qualities are very important for user experience, while pragmatic qualities are a segment of user experience. When making choices, users usually highlight pragmatic qualities in their reasoning over hedonic qualities because they are easy to justify, while hedonic qualities are the ones generating pleasant experiences (Diefenbach & Hassenzahl, 2011).

In an earlier study, Hassenzahl, Platz, Burmester and Lehner (2000) researched and contrasted ergonomic and hedonic qualities. Ergonomic qualities are paralleled with usability and task-related functions (Hassenzahl et al., 2000). Thus, ergonomic and pragmatic qualities have many similarities and could be even be seen as two terms for the same matter. Pragmatic qualities as a term better represents the task-focus and is therefore a better term when being contrasted to hedonic qualities. Ergonomic qualities have been used in earlier studies (Hassenzahl et al., 2000) while pragmatic qualities are being used in more recent studies (Diefenbach & Hassenzahl, 2011; Hassenzahl et al., 2010). Hassenzahl et al. (2000) observed that ergonomic and hedonic qualities can be independently perceived by users and that ergonomic and hedonic qualities are negatively dependent of each other in several ways. E.g. if a software is too easy to use, it cannot be fun to use (Hassenzahl et al., 2000). However, both qualities are important when assessing the appeal of software (Hassenzahl et al., 2000).

In a later study, Hassenzahl et al. (2010) discovered that these two qualities, pragmatic and hedonic, affect differently when evaluating experiences. Hedonic qualities are "motivators", which produce positive experiences by fulfilling needs while pragmatic qualities are "hygiene factors" that dampen negative effect by removing barriers but are not sources of positive experiences in themselves (Hassenzahl et al., 2010). Additionally, they noted that experiences can be evaluated and divided into different categories (Hassenzahl et al., 2010). Benefit being that when describing a product, it would be easier to describe experiences with it instead of the product itself (Hassenzahl et al., 2010).

3.5 Components of user experience

There are differing perspectives on the components of user experience. One option is that user experience consists of user, product and context of use (Forlizzi & Ford, 2000; Hassenzahl & Tractinsky, 2006). These can also be seen in the definition given by the latter study (Hassenzahl & Tractinsky, 2006). Roto et al. (2011) have similar categories of context, user and system still a few years later. However, in a newer study conducted by Tokkonen and Saariluoma (2013), by interviewing experts they perceived that the main components of user experience are user, product and company. They note that previous earlier studies have slightly differing results and mention that they see context as being included in product use (Tokkonen & Saariluoma, 2013). They also noted that they found several approaches, concepts and definitions for user experience (Tokkonen & Saariluoma, 2013). User and product appearing in both perspectives are both present directly or indirectly in the definitions given in chapter 3.3.

Developers with different backgrounds understand user experience in different ways (Tokkonen & Saariluoma, 2013) while academics understand it in a yet another way (Hassenzahl, 2008). Reasons for these differences include user experience being "associated with a broad range of fuzzy and dynamic concepts" (Law et al., 2009).

3.6 Anticipation and expectations

Anticipated use in user experience is relatively rarely researched in comparison to during and after use (Bargas-Avila & Hornbæk, 2011). Nevertheless, an example of a user's internal state in the definition for user experience by Hassenzahl and Tractinsky (2006) was expectations. ISO-standard (International Organization for Standardization, 2010) for user experience also mentions anticipated use. Expectations and anticipation are central to user experience. McCarthy and Wright (2004) see anticipation as a part of how people make sense of experiences. Anticipated user experience applies to the period before use, but also from the momentary user experience of a single moment to the cumulative user experience of the system after using it for a while (Roto et al., 2011). Having used a system before affects how we anticipate future interactions to go (Roto et al., 2011). However, Roto (2007) sees experience preceding interaction with a product not as user experience, but as expected user experience. It still plays a key role in her view when it is being evaluated against user experience after interaction has started (Roto, 2007). This is consistent with how anticipated user experience impacts user experience. Expected user experience could be seen as one part of anticipated user experience, focused on the time before the first interaction.

People expect their experiences to be greater than what they will be in reality (Yogasara, Popovic, Kraal, & Chamorro-Koc, 2011). With innovative products, there is a risk of users having expectations that do not match what the product is capable of in reality (den Ouden, Yuan, Sonnemans, & Brombacher, 2006). These expectations vary between different user types of novice, occasional and experienced users (den Ouden et al., 2006). IoT is an innovation fitting this description and as such, user expectations of IoT usage need to be managed. When expectations are not met, this can lead to frustrating situations. When dealing with computer use, user frustration is a serious problem (Lazar, Jones, & Shneiderman, 2006). Frustrating encounters with computers can waste close to half of the time spent working on one (Lazar et al., 2006). Addition of Internet functionality to everyday things could very well lead to increased frustration when using such things, due to e.g. aforementioned asynchronicity or connection problems, things that are not necessarily expected of everyday things. When devices are functioning according to its specification but not functioning according to the user's expectations, this can be seen as a failure on the user's end (den Ouden et al., 2006).

Olsson (2014) presents a work-in-progress framework of user expectations. It consists of desires, experience-based assumptions, social & societal norms and must-be expectations (Olsson, 2014). Desires reflect what people truly value technology to offer based on inherent human needs, values, attitudes and personality (Olsson, 2014). Experience-based assumptions are based on user's own experiences and other significant people and expresses people's habits and how they conceptualize technology to perform, behave and evolve (Olsson, 2014).

Unlike desires, which generally refer to positive expectations, experience-based assumptions can also be negative (Olsson, 2014). Social and societal norms depict what people, irrespective of their own desires or prior experiences, assume technology to allow based on what phenomena and trends currently hold (Olsson, 2014). Must-be expectations represent minimum requirements for user acceptance that any new technology should have, using personal long-term experience of different types of products and technologies as a foundation (Olsson, 2014). These are referred to as hygiene factors by Olsson (2014), similar to how Hassenzahl et al. (2010) refer to pragmatic qualities of user experience. These "hygiene factors" are very applicable to IoT, such as with asynchronicity. Also, sometimes, the lack of an experience is a good user experience, such as with elevators (Rousi, 2014). A good elevator ride goes unnoticed, while a bad ride leaves the user with a negative experience (Rousi, 2014). A thermostat that does not turn up the heat fast enough leads into a bad experience while a comfortable temperature goes unnoticed.

3.7 Evaluating user experience

User experience is dynamic, because a person's internal and emotional state is ever-changing, and there are differences in circumstances both during and after interaction with a product (Vermeeren et al., 2010). Therefore, in addition to evaluating user experience after interaction, it should also be evaluated before and during (Vermeeren et al., 2010). Long-term user experience evaluation is generally seen as interesting, relevant and useful by both developers and managers (Varsaluoma & Sahar, 2014). There are many methods for conducting user experience evaluation. Vermeeren et al. (2010) gathered a total of 96 user experience methods over a span of several years. The study conducted by Alves et al. (2014) discovered that the most common methods are:

- Observation
- Think Aloud
- Contextual Interview/Inquiry
- Interviews
- Experience prototyping

In addition to these, they also noted that when a software designer, with their expertise being in software engineering, evaluates user experience, they are likely to use other software engineers to do their user experience evaluations (Alves et al., 2014). They also noted that end users are used to assess user experience in less than 50% of cases (Alves et al., 2014). In contrast, researchers seem to favor questionnaires (Bargas-Avila & Hornbæk, 2011). Experts can evaluate user experience with no issues, but they have problems adopting the perspective of the user (Lallemand, Koenig, & Gronier, 2014).

3.8 Critique and limitations

User experience as a field of study and a framework has also received critique and it has shortcomings. It has been called “vague, elusive and ephemeral” (Hassenzahl & Tractinsky, 2006). Measurability is also a problem with some researchers defying measurability while others embrace it (Law, 2011). Other limitation of the field is the lack of an established definition (Law et al., 2009). Having a shared definition would help establish an integrated framework of user experience (Law, Roto, Vermeeren, Kort, & Hassenzahl, 2008).

3.9 Programmers as users

There seems to be a sparse amount of research on programmers as users, with research instead focusing on end user programmers (e.g. Prabhakararao et al., 2003). In 2005, it was estimated that there would be over 13 million self-reporting end user programmers in the US by 2012, compared to 3 million professional programmers (Scaffidi, Shaw, & Myers, 2005). Additionally, there would be many spreadsheet and database users that do not identify themselves doing programming, but still do that to some extent (Scaffidi et al., 2005). Based on these figures, it is understandable that research has focused on end user programmers. However, it is the professional programmers that enable end user programmers alongside other end users. Chapter 4.1 mentions that controlling IoT devices remotely and automatically are programming-like activities (Rowland, Goodman, Charlier, Light, & Lui, 2015). It is possible that improving the user experience for professional programmers would enable them to improve the work they do on IoT alongside other fields. To do this, programmers would need to be seen as users themselves. This is an area of research that could warrant further research.

4 USER EXPERIENCE OF INTERNET OF THINGS

This chapter explores the topics of designing user experience for Internet of Things and design principles and guidelines for Internet of Things systems.

4.1 Designing user experience for Internet of Things

Designing Internet of Things devices and services has different user experience design challenges from traditional digital services (Rowland et al., 2015, pp. 4). Consumer IoT differs in ten ways from "conventional" user experience according to Rowland et al. (2015, pp. 28): 1) asynchronicity and discontinuity; 2) latency; 3) code can be run in many places; 4) devices being distributed in the real world; 5) functionality can be distributed across multiple user interfaces; 6) much of the information processing is done in the Internet service; 7) controlling devices remotely and automatically are programming-like activities; 8) differing technical standards; 9) possibility of complex services being used by many users over many user interfaces, devices, rules and applications; 10) IoT is enabling collecting and acting on data that has not been available previously.

The first way is that embedded devices often connect only intermittently to save power, leading into asynchronicity and discontinuities in user experience (Rowland et al., 2015, pp. 8-9). Power saving has been elaborated on in chapter 2.4, while asynchronicity ties into the second way. The second way is that even though we expect physical things to respond immediately, latency on the Internet is out of your control while reliability is not an absolute either (Rowland et al., 2015, pp. 7-8). Latency and reliability can affect design decision such as how to represent sent commands in user interface (Rowland et al., 2015, pp. 62-65).

The third way is that code can run in many places, meaning that if a part goes offline, the user must engage with the system model to predict how it will work in such a situation (Rowland et al., 2015, pp. 9-11). This requires more effort from the user (Rowland et al., 2015, pp. 9-11). The fourth way is that because the devices are distributed in the real world, the social and physical context of use is complex and varied (Rowland et al., 2015, pp. 11). The fifth way is that functionality can be distributed over multiple user interfaces, meaning that interusability needs to be considered alongside usability (Rowland et al., 2015, pp. 5). This means that user experience needs to be coherent across all devices that the user interacts with (Rowland et al., 2015, pp. 337).

The sixth way is that much of the information processing occurs in the Internet service, often making the service experience equally or more important than user experience of an individual device (Rowland et al., 2015, pp. 6-7). The focus of user experience might be in the service, while the devices are interchangeable (Rowland et al., 2015, pp. 6-7). The seventh way is remote control and automation being programming-like activities, leading into IoT breaking the concept of direct manipulation of user interfaces (Rowland et al., 2015, pp. 11-13). Direct manipulation gives users direct feedback, while configuring an IoT service requires anticipating future needs, which is harder cognitive task (Rowland et al., 2015, pp. 11-13). The eighth way is the many differing technical standards, leading into different interoperability problems (Rowland et al., 2015). This is also elaborated on in chapter 2.3. The ninth way is that complex services can have many devices, users, user interfaces, rules and applications (Rowland et al., 2015, pp. 13-14). This leads into it being very difficult to understand and manage the interrelations of different services and devices (Rowland et al., 2015, pp. 13-14).

Finally, the tenth way that the user experience of IoT differs from that of other systems is that IoT enables capturing and acting on unprecedented data, which needs to be used as design material by designers (Rowland et al., 2015, pp. 16-17). This data can be used to design and deliver better service (Rowland et al., 2015, pp. 16-17). Ownership of this data and privacy concerns are a possible issue with this, though (Hurlburt et al., 2012). Thus, whereby user experience is already a complex domain, IoT presents an interactive system in which user experience can be felt and is affected by numerous levels of information, devices and operations. This can be confirmed by Lee, Prenzel and Bien (2013), who have advocated for specific design principles to be used when designing for the IoT.

4.2 Design principles and guidelines for IoT systems

When designing an IoT system that is wholly controlled by a development team, interoperability and privacy, the top two challenges presented in TABLE 1 on page 11, are easier to control. When the system is not controlled by the team, these challenges become more difficult to manage. The system needs to be interoperable with other relevant systems, being able to both take advantage of data collected from external systems and produce usable data for these systems to make use of. All this operates in a way that protects the privacy of the user and the objects in the system. Users might expect for all the systems to interoperate seamlessly and without a hitch (Hurlburt et al., 2012). However, this might not happen due to unforeseen circumstances (Hurlburt et al., 2012).

Interoperability between systems might not be enough. A system with highly flexible application scenarios still needs to understand what the user might need from the system in varying contexts (Lee et al., 2013). For example, in an in-home care scenario, when the user wants to go out, they might want the system to heat coffee if going out for a recreational activity but not before a medical exam (Lee et al., 2013). Another user might not drink tea instead of coffee. This is the reason why the user and associated use scenarios in the IoT design process are an integral component.

Lee et al. (2013) offer four principles as guidelines for designing user-centered learning IoT systems. The first principle is that even with multiple learning strategies, the system should rely on a specific learning strategy if it strongly believes about what happens next or what to do next, based on the observable representation of what it predicts about state and outcome (Lee et al., 2013). Probability of the best possible outcome should be encoded by the computational model alongside a quantification of corresponding uncertainty (Lee et al., 2013). For the second principle, the system should have an inherent inclination to subdue strategies that are risky and have high demands (Lee et al., 2013). Instead, it should favor safe and easy strategies (Lee et al., 2013). The third principle is that the system should be able to switch between several learning strategies for situations where previous choices have been unsuitable, but the user is still hesitant to choose a new strategy (Lee et al., 2013). The fourth design principle is that switching strategies should also incorporate the possibility of the user switching the strategy, allowing for a push-pull mechanism (Lee et al., 2013).

5 RESEARCH METHODS

This chapter explains how the questionnaire was developed, how participants were recruited and what ethical considerations there have been related to this paper.

5.1 Developing the questionnaire

A test run of the first version was done by adapting a combination of second and third phase questions from the longitudinal student survey created by Myllärniemi, Kujala, Raatikainen, & Sevón (2016). The original survey was implemented in three phases: in beginning of the course capture expectations, 3 weeks from the project start for initial experiences and at the end of the 3-month project for final experiences (Myllärniemi et al., 2016). The test run questionnaire for this thesis had open questions and questions on a Likert scale of 1-5 on two different pieces of paper, forcing the participant to switch between the two forms. The second version had the addition of expectations to many of the open questions (e.g. questions 16 and 17 in TABLE 3) and more instructions to the topics based on explanations given in the survey by Myllärniemi et al. (2016). Explanations of software frameworks were written by the researcher. A question about development tools used was added (question 19 in TABLE 3), alongside questions on a Likert scale on different areas meeting their expectations (questions 18, 23 and 28 in TABLE 3). The Likert scale was changed from 1-5 to 1-7. This is how it was also in the Myllärniemi et al. (2016) survey. The two types of questions were combined to a single continuous form. While the original Myllärniemi et al. (2016) survey was done over the Internet, this questionnaire was done in person. Audio was also recorded with permission (see Appendix 1) and the participants were encouraged to speak out, especially if they could not form their thoughts as text. Notes were taken on the time of recording and what answer they were talking about.

This second version was also completed by one participant and was edited with some small additional modifications. These changes were on clarity of the form and adding more emphasis in text to expectations. Otherwise the second version was identical to third in the content of questions and as such, the one framework filled on the second version was used in the final results. The third, final version (see Appendix 2) was filled on a computer, instead of by hand. This removed the need to write up the answers and the possibility ambiguity in interpreting handwriting. The questions were given out to participants beforehand to give them time to think on them.

The answers given to the single run of the first version was later expanded upon by having the participant fill out the missing and changed questions by themselves on their own time. This was done to gain an additional needed answer. This was not done in person, because the participant was not available to be met at that point.

TABLE 3 below describes the questions in the questionnaire in English. Most of the questions were adapted from Myllärniemi et al. (2016). Questions were asked in Finnish from the participants, as seen in Appendix 2. Questions with (Likert) after them were on a 1-7 Likert scale while the rest were open questions. Questions 10 and 12 asked to answer on negative experiences. These results were turned over for calculating means and standard deviations.

TABLE 3: Questions in the questionnaire

Question
1. Describe the usage of the framework in IoT usage.
2. Your experience with similar technologies.
3. Estimate how many hours you **expected** to spend installing or finding out information related to installing the framework and how many you spent.
4. Estimate for how many hours you have used the framework.
5. Framework feels good. (Likert)
6. I enjoy using the framework. (Likert)
7. Using of the framework is rewarding. (Likert)
8. If you wish, tell more on your experiences regarding the questions 5-7.
9. Framework fulfills my requirements. (Likert)
10. Using the framework is frustrating. (Likert)
11. The framework is easy to use. (Likert)
12. I need to spend too much time correcting things when using the framework. (Likert)
13. If you wish, tell more on your experiences regarding the questions 9-12.
14. Framework meets my expectations. (Likert)
15. APIs support completing the task. (Likert)
16. Describe what problems you **expected** from the APIs and what problems you ended up having.
17. Describe what strengths you **expected** from the APIs and what strengths it

ended up having.
18. APIs meet my expectations. (Likert)
19. What development tools you have used for working with the framework?
20. Development tools support completing the task. (Likert)
21. Describe what problems you **expected** from the development tools and what problems you ended up having.
22. Describe what strengths you **expected** from the development tools and what strengths it ended up having.
23. Development tools meet my expectations. (Likert)
24. What information sources you **expected** to use related to the framework and what information sources you have used.
25. Information sources support completing the task. (Likert)
26. Describe what problems you **expected** from the information tools and what problems you ended up having.
27. Describe what strengths you **expected** from the information sources and what strengths it ended up having.
28. Information sources meet my expectations. (Likert)
29. Your own comments on the topic. Any thoughts that come to mind.

Questions 1 and 2 were added to give insight to the framework used and on how experienced the participant was. Questions 3 and 4 are adapted from questions on usage and questions 5-8 were adapted from questions on Enjoyment from Myllärniemi et al. (2016). Enjoyment was used by Myllärniemi et al. (2016) as a measure on how intrinsically motivating the framework was. Questions 9-13 were adapted from questions on Usability from Myllärniemi et al. (2016). They used Usability Metric for User Experience (UMUX) as a basis to measure subjective usability (Myllärniemi et al., 2016). Besides questions 14, 18, 23, 28 and 29, rest of the question were adapted from Myllärniemi et al. (2016) theme of "Support from platform boundary resources". They had been identified as key means for facilitating application development (Myllärniemi et al., 2016). These can be further broken down to topics of application programming interfaces (APIs), development tools and information sources (Myllärniemi et al., 2016). Questions 14, 18, 23 and 28 were added to gain insight into how the framework and the platform boundary resources met each participant's expectations. Question 29 was added to collect possible insights on the topic the participants might have had.

5.2 Recruiting participants

The questionnaires were filled by students of Jyväskylä University of Applied Sciences (JAMK). These students had been working on practical work training, developing a proof-of-concept prototype of an IoT-service for sewer and drain water networks. They were participants with little previous experience in development and who had just started working with IoT some weeks earlier. As such, they could recall or at least give estimates of what they had expected of the different areas.

The third version of questionnaires were filled during the penultimate week of the course and the IoT development project. This was done to leave some time for any follow-up questions or clarifications. By then, they had done most of the work and were finishing the project. In total, there were seven participants. One of the participants answered on three different frameworks while an another answered on two. This brings the total filled questionnaires to ten.

The questionnaires were filled in a classroom next door to the one they were working at. Dates and times of interviews were agreed upon a day or two earlier, having generally either one or two participants filling the questionnaire each day. This was done to give time in the afternoon to listen to the recordings and write them down while the memory was still fresh, and it was possible to remember what they had been talking about.

5.3 Ethical considerations

Short verbal instructions (see Appendix 3) were given to the participants, that also included information about how their information and answers will be handled. Their right to refuse audio recording was mentioned as well. Written consent (see Appendix 1) was acquired from the participant before starting each interview. This included information on what the study was about, on the procedure, mentions that they will not be harmed by participating to the study, that they can stop at any time and that the results will only be used for scientific reporting in a way that individual participants cannot be identified. Answers were mentioned to be handled confidentially, anonymously and as safely as possible. Two copies were signed by both the participant and the researcher. Each participant was given one of the copies. Written consent appended here as Appendix 1 has contact information retracted.

Names of the participants were collected on a background information form (see Appendix 4) for possible latter clarifications and so that they could check their answers if they wanted to. This latter possibility was mentioned in both the written consent and verbal instructions. Answers given in the background information form were not used in this paper. All participants agreed for the audio of their session to be recorded. None exercised their right to stop the session. There is a possibility that a person, whom is familiar with the practical work training course, could identify some of the participants from this paper based on the frameworks they had been working with.

Permission to use the survey by Myllärniemi et al. (2016) as the basis for the questionnaire in this paper was asked for from the authors.

6 RESULTS AND DISCUSSION

This chapter gives information on the frameworks that the participants had been using, what kind of expectations each participant had for different areas related to the frameworks, discussion on questions, results related to the research questions, discussion on the form and procedure and limitations of the research.

6.1 Frameworks

The participants answered on eight different frameworks: Ionic[1], Angular2[2], BaasBox[3], Java API for RESTful web services (JAX-RS)[4], Python[5], Robot Framework[6], Docker [7]and Kaa[8]. They all worked on the same project in different roles.

Ionic framework is used for building mobile applications. Angular2 is a framework used for developing both mobile and desktop applications. BaasBox is an open source backend software. JAX-RS API is used for creating web services. Python is a programming language. Robot Framework is used for test automation. Docker is a platform for software containerization. Kaa is an IoT middleware.

Of all the frameworks, only Kaa is specific for IoT development. However, all the other frameworks are used for the full stack of technology required to make an IoT service. As such, the other frameworks, even if not IoT specific, are still an integral part of the whole IoT service.

[1] https://ionicframework.com/
[2] https://angular.io/
[3] http://opensource.baasbox.com/
[4] https://github.com/jax-rs
[5] https://www.python.org/
[6] http://robotframework.org/
[7] https://www.docker.com/
[8] https://www.kaaproject.org/

6.2 Expectations for each participant

Out of all questions, there were four (one in each group) that were on how the framework and different related platform boundary resource topics met their expectations: the framework itself (question 14. in Appendix 2), APIs of the framework (question 18. in Appendix 2), development tools used in developing with the framework (question 23. in Appendix 2) and information sources related to the framework (question 28. in Appendix 2). Each of these were on the Likert scale of 1-7 (from fully disagree to fully agree) in TABLE 4 below. The questions did not specify if the expectations should be positive or negative, but the participants tended to reflect through positive expectations on these questions.

TABLE 4: Results grouped by respondent

Participant	Framework	Mean	Standard deviation	Expectations
1	Ionic	6,642	0,633	6-6-7-6
1	Angular2	5,286	1,541	6-6-6-5
1	BaasBox	6,643	0,745	7-7-7-5
2	JAX-RS	5,857	1,027	6-6-6-3
3	Python	5,714	1,139	7-4-7-5
4	Robot	6,500	0,941	7-7-7-7
4	Docker	6,213	0,893	7-7-6-7
5	Kaa	4,714	0,994	5-5-6-5
6	Kaa	5,071	1,072	5-6-6-6
7	Angular2	5,429	1,399	6-6-7-5

Each participant had their own expectations of their respective frameworks. The first participant answered on three different frameworks. These were Ionic framework, Angular2 and BaasBox. For Ionic framework, they answered 6 on how the framework met their expectations, 6 on how the APIs met their expectations, 7 on development tools and 6 on the information sources. Over all questions, the mean for this framework by this participant was 6,642 with a standard deviation of 0,633. Out of all surveys, this had the third highest mean with the lowest standard deviation. Only answer below a 6 was in question 10, do they find the framework frustrating to use. They had had some similar previous experience.

For Angular2, this participant 6, 6, 6 and 5 to the four questions on each area meeting their expectations. Mean for the answers was 5,286 with a standard deviation of 1,541. The lowest answers were on frustration and wasting time. Strongest areas were APIs and tools. The participant had some similar previous experience.

For BaasBox, the first participant answered 7, 7, 7 and 5 for expectations. Mean for their answers on BaasBox was 6,643, second highest, with a standard deviation of 0,745, second lowest of all. The lowest answers were for the two questions in information sources. Highest answers were on APIs and tools with maximum points while the framework missed this by one point. This survey had the highest mean with the second lowest standard deviation. The participant had no previous experience with similar products. They had previously done some simple solutions themselves.

The second participant answered on one framework, JAX-RS. For expectations, they gave 6, 6, 6 and 3 with a mean of 5,857. Standard deviation was 1,027. Lowest answers were on how information sources met their expectations. The highest answers on an area were for development tools. They had some similar previous experience.

The third participant answered on Python as their framework. Python is a programming language, but they were not able to pinpoint any one or two frameworks as such. Instead, they coded in Python with the help of some modules. They answered on expectations with 7, 4, 7 and 5. Mean of all answers was 5,714 with a standard deviation of 1,139. Lowest answer regarding points was on frustration with the framework, which they somewhat agreed with. Highest points for an area was for development tools. They had no previous experience in this topic.

The fourth participant answered on Robot Framework and Docker. For Robot, they answered with 7, 7, 7 and 7 on expectations with a mean for all answers being 6,500 and a standard deviation of 0,941. Lowest answer was for how the APIs support the task at hand. They viewed the files as the APIs and were uncertain if writing to files was a strength or not. Tools and information sources were both areas with maximum points on the scale. They had no previous experience.

For Docker, the fourth participant answered on expectations with 7, 7, 6 and 7. Mean of all answers was 6,213 with a standard deviation of 0,893. Lowest answer was again for how the APIs support the task at hand. For this framework, they saw the command line interface and files as the APIs. However, this time they did know to expect ease of use. Maximum points were given for information sources. They had had previous experience with virtual machines, which have similar functionality.

The fifth participant answered on Kaa. Answers to expectations were 5, 5, 6 and 5. Mean for all answers was 4,714. This is the lowest mean out of all answers. Standard deviation was 0,994. Lowest answers were on ease of use and not having to spend too much time fixing things while using the framework. Area with highest points was development tools. No answer was given full points. They had no previous experience.

Also, the sixth participant answered on Kaa. They answered on expectations with 5, 6, 6 and 6. Mean of all answers was 5,071, second lowest of all answers. Standard deviation was 1,072. Lowest points as an answer was to the questions of if the framework fulfill their requirements and if using the framework is frustrating. Highest points for an area was for the three areas of APIs, development tools and information sources. No answer was given the full points. They had no previous experience.

The seventh participant answered on Angular2. For expectations they answered 6, 6, 7 and 5. Mean of all answers was 5,429 with a standard deviation of 1,399. Lowest answers were on if the framework does feel good and if it fulfills their requirements. Highest points for an area was for development tools. They had previous experience with different frontend frameworks.

6.3 Discussion of questions

TABLE 5 below shows the mean and standard deviation for each Likert scale question seen in Appendix 2 across all participants. These are expanded on after TABLE 5.

TABLE 5: Results grouped by question

Question	Area	Mean	Standard deviation
5.	Framework	5,5	1,179
6.	Framework	5,6	1,075
7.	Framework	5,9	0,876
9.	Framework	5,8	1,398
10.	Framework	4,7 (inverse)	1,636
11.	Framework	5,7	1,567
12.	Framework	5,2 (inverse)	1,751
14.	Framework	6,2	0,789
15.	APIs	6,0	1,155
18.	APIs	6,0	0,943
20.	Development tools	6,8	0,422
23.	Development tools	6,5	0,527
25.	Information sources	6,0	0,943
28.	Information sources	5,4	1,160

On question 2 (previous experience), half of the participants had no experience with similar technologies, while none but one had experience with the specific technology used here. Lack of experience makes sense, as the participants were students participating in practical work training. Additionally, many of the technologies used for the project were very new and still under development.

On question 3, setup time estimations matched for only one participant, while other estimations were inaccurate. More than half of participants overestimated their setup times. It seems that students might expect installation of new development software to take longer than in reality.

Question 4 had framework had most of the usage estimates ranging from 30 hours to 150 hours, with a single clear outlier of 400 hours. This outlier was for Python. Unlike other frameworks, Python is a programming language, thus possibly explaining some of the deviation from other answers. The participant also said that they had familiarized themselves with it for some amount at home. From the answers and comments, it was not revealed if they had used Python before the course, but this is a possibility.

While for questions 5 (does it feel good?) and 6 (enjoyment), Kaa and Python got 4s as answers, in question 7 (does it feel rewarding?) nothing went below 5. For each of these questions, the average answers were within 0,4 of each other, but question 6 had the highest mean (5,9) with the lowest standard deviation (0,876) of the group. The frameworks were found slightly more rewarding, but they were still quite enjoyable and felt good.

Four of the participants mentioned a learning curve in question 8, in relation to previous three questions. If these questions would have been asked when the participants started using the frameworks, responses might have been more negative with the learning curve still being climbed. Two of the participants mentioned that their framework still being incomplete brought their own problems. A participant answering on Kaa mentioned, that there was still missing functionality. A participant answering on Angular2 mentioned modules being targeted for different release candidate versions and having major changes between them.

For question 9, requirements seem to be fulfilled worst by still incomplete frameworks. This makes sense, because they are not fulfilling all the requirements set by the developers to be considered a fully functional framework.

On question 10 (frustration), Angular2 had two participants, one rating it 6 and one as 3. While 3 is barely below mean for this question, 6 is the highest of all answers and thus most frustrating. Because this question asked about negative emotions, frustration, 6 compares to a 2 in other questions. This question had the lowest mean (4,7), when turned around. It also had one of the highest standard deviations (1,636). While Angular2 got a 6, JAX had a 1. Because we are comparing individual answers here, some of this might be explained by the particular participants. However, the same participant that gave Angular2 a 6 on this question, has given positive answers elsewhere, so they do not seem to be inclined towards negativity. Yet another participant gave Angular2 a 3, which is not the lowest but towards the lower end. They did say aloud that they are a positive person and that is why things do not frustrate them. Therefore, Angular2 might have caused frustration to this person as well, if they were more liable to frustration.

Continuing that, for question 11 (ease of use) this same person gave Angular2 the value of 6, calling it easy to use, while the other person gave it a 3, adding in question 13 that it is not always intuitive to a beginner. Out of all frameworks, incomplete ones were generally found the hardest to use. This seems to be the case in question 10 as well. Lack of documentation and help might be contributing to these feelings.

Question 12 (spending time fixing things) had the highest standard deviation (1,751) of all answers, even if the reversed mean (5,2) is not as high as the one in question 10. Both questions dealing with negative feelings have the lowest means and highest standard deviations. This gives reason to think that they are not fully comparable to others.

Question 13 added some more context to others. Angular2 alongside Kaa, which was also ranked quite low in these questions, were mentioned as not being easy to use for beginners. In question 14 (expectations), Kaa was ranked as lowest for meeting expectations. It was the only one below 6. In addition to a learning curve, some functionality was still missing and contributed to this result. Otherwise, the question had a high mean of 6,2. It is the only question related to the frameworks themselves that has a mean that is above 6.

Questions 15-18 were concerned with APIs. For question 15 (supports task), the same participant answering on two different frameworks (Robot Framework and Docker) answered with 4s, saying in questions 16 (problems) and 17 (strengths) that they had no expectations for strengths or weaknesses. Yet in question 18 (expectations) they gave both frameworks 7 for meeting expectations. There seems to be a slight contradiction here. Part of this problem might be the wording of the question 18 (expectations). More on this in chapter 6.5. While person answering on Python gave 4 in question 18, because they did not expect any strengths or weaknesses. Question 18 and 15 had the same mean (6,0), but the former had a lower standard deviation (0,943).

In question 16, four did not expect problems, of which two got some problems. A participant answering on Kaa did not really expect problems, but had some issues, using poor REST-API documentation as an example. Participant answering on Python mentioned learning something new as the only problem. Five participants expected problems but got different problems instead. aside from one participant. Participant answering on Ionic expected starting to use the framework as being difficult, but it was not. They encountered other problems but did not specify. A participant answering on Angular2 expected the documentation to be difficult to understand, but it was understandable. A problem was that it was still incomplete. Other participant answering on Angular2 expected passing of information to be tricky, but it was easy. Instead, it was so easy that they were worried about security. Participant answering on JAX-RS expected it to be hard to install, but installation was easy thanks to plugins for the integrated development environment (IDE). Instead, they had problems with HTTP-methods. Participant answering on BaasBox expected using the API to be difficult, but it already had most of what they needed and creating new ones was easy. They did not mention having any problems. One participant got the problems that they expected, which was lack of help for certain things for Kaa. Clearly, API related problems were difficult to expect right.

In question 17, strengths were easier to expect, with three cases of strengths being expected. Two of these, one for Angular2 and other for Docker, were on ease of use while the third was also for Angular2 but did not specify what strengths they meant. Three expected more problems than what they got, which can be seen as a strength that was not expected. These were for Ionic, BaasBox and JAX-RS. Three did not expect or could not name any strengths due to inexperience. These were for Python, Robot Framework and Kaa. One answer on Kaa was unclear.

Questions 19-23 were about development tools. In question 19 (tools used), a few of the participants prefer NetBeans over Eclipse as tools for web development. Atom, a text editor, was new for almost all participants, but was well regarded. Tools for all frameworks, except for Kaa, were given 7's in question 20 (supports task) for supporting the task being undertaken. Kaa got 6 from both participants. This question had the highest mean (6,8) and lowest standard deviation (0,422) out of all questions.

As seen in answers for question 21, there were no great problems with tools. Six participants did not expect any problems but got some small ones. These were for Ionic, BaasBox, Python, Robot Framework, Docker and Kaa. Four did expect something small, of which three got it right. These three expected problems were Git merge problems for a participant answering on Angular2, problems with finding module versions for other participant answering on Angular2 and small IDE problems for a participant answering on Kaa. Participant answering on JAX-RS did not have problems they were expecting but had some other small problem.

While in question 22 (strengths), four had their expectations being met and three had their expectations exceeded. These exceeded expectations were for Atom, browser development tools, Git and NetBeans. Two of the remaining three answering on BaasBox and Python did not expect anything and could not name anything while the participant answering on Ionic did not expect anything but found strengths with Atom and browser development tools.

Expectations of development tools were met well based on question 23. Ionic, BaasBox, Python and Robot Framework got 7s, Angular2 got a 6 and a 7 while the rest got 6s. This question also had a high mean (6,5), low standard deviation (0,527) and no answer below 6, like question 20. Just not to the same extent. This section had the highest mean and lowest standard deviation for met expectations. Based on these results, it could be said the participants are most content with development tools. Tools used for each task are in almost all cases something that is used not just for developing with the specific framework, but also for other similar tasks. Tools used here are not software that is still in their beta-phase, but generally, something established and extensively used in wider circles.

Questions 24-28 were about information sources. In question 24 (expected and used sources), five used what they expected while four also used additional sources on top of what they had expected. Six participants used Google search, as they had expected to be using. Seven participants used documentation, of which participant answering on Robot Framework had not expected to use it. Other expected information sources were StackOverflow and forums, with both having three mentions by different participants. StackOverflow can be seen as type of a Questions & Answers forum and might be included for the participants giving forums as an answer. Three participants used blogs and tutorials unexpectedly. Participant answering on Python used original open source code without having expected using them. One of the participants answering on Kaa expected to use wikis, which they did. A participant answering on Angular2 used the GitHub code repository, from where they expected to use the original open source code, wikis in the repository and readme documentation. Additionally, they found themselves using the issues section of the repository. Participant answering on JAX-RS had not used all the sources they had expected to use. They ended up only using official documentation because they found everything else to be lacking.

Information sources supported development in question 25 (supports framework) with a mean of 6 and a low standard deviation (0,943). Nothing was below 5. Still some beta-phase documentation was still incomplete in places, especially with Kaa, Angular2 and plugins for BaasBox. For some modules, Python also had lacking documentation. In general, established frameworks had the most developed documentation, as is to be expected.

Half of the participants had unexpected problems with their information sources based on answers to question 26. Three had the problems they had been expecting, while one of these also had some additional problems. Both participants answering on Kaa expected there to be little or incomplete documentation, which came true. A participant answering on Angular2 expected themselves not to be capable on using Google, which came true. In addition, they were surprised that they had to go to the issues section of the GitHub repository, which they had never done before. Other participant answering on Angular2 expected documentation to be complex, but it was understandable. The problem was that it was incomplete and that some documented solutions were no longer functional with newer versions. Participant answering on BaasBox did not expect problems, but documentation was not always easy to use and there were little discussion and solutions to problems online. Participant answering on Ionic had the opposite problems to what they had expected. They had expected there to be too much documentation, but it was the opposite in some cases. For JAX-RS, the participant expected documentation to be complex, but the problem was finding them. Participant answering on Python did not expect problems but finding examples for some modules was difficult. Participant answering on Robot Framework expected no one to have the same problems, but they found out that the problem was the sheer length of documentation. Participant answering on Docker did not expect any problems, because in theory the tools should not have problems with different setups.

For strengths in question 27, four participants had their expectations being met. Participant answering on Ionic expected the documentation to be precise, which it was. Not all of it was easy to read, however. For both Robot Framework and Docker, the participant expected and found examples for getting started. A participant answering on Kaa expected there to be answers to most common problems, which there were. However, for rarer cases they had to ask for help. Two participants had their expectations be met partially. Participant answering on BaasBox expected documentation to be easy to read, but it was somewhat unclear because of their own inexperience. Other participant answering on Kaa expected to find professional wiki-pages, but some areas were still missing or incomplete. Two participants did not have their expectations being met. Participant answering on Python expected there to be a wide variety of examples, but they were lacking for the parts they needed. Otherwise the documentation was satisfactory. A participant answering on Angular2 expected up-to-date information, but it was for different versions and did not necessarily work with what they were using. A participant answering on Angular2 did not have expectations, but found documentation to be surprisingly straightforward, if not comprehensive. Participant answering on JAX-RS did not expect or name any strengths.

In question 28 (expectations), information sources did not meet the expectations quite to the same extent as others, having the lowest mean of 5,4 with the highest standard deviation of 1,160. Because many of the frameworks were in beta, there were not as many information sources available. Documentation was partially incomplete and there were little conversations on the topics to draw expertise from. However, JAX, which is an established framework, had the lowest answer of 3 in this question. They had problems finding the official documentation at first and could not find some things that they needed anywhere.

Question 29 asked for comments on the topic. For Ionic, the participant found it good to work with. Their own needs were not very diverse, but it should work with more ambitious projects, at least if they use Angular. They thought that Ionic can be used to speed up smartphone application development with techniques familiar from web-development. For Angular2, they thought it seemed like a good tool for the future. At that point, however, using it was still unpredictable. Other participant answering on Angular2 noted that using a beta phase framework is nice as a hobby, but that you should not use one in a real system. You would have to capsulate and tie to certain versions and live with the limitations that come with them. The participant answering on BaasBox found BaasBox to be a good framework because it simplifies managing a system considerably. They might use it later. The participant answering on JAX-RS had used REST APIs previously and were interested in them. They noticed that Linux works better for web development than Windows. They had also heard about security issues with Java, but they used it nonetheless. The participant answering on Python found Python to be good and fitting for beginners. However, they would not use it themselves in the future, because they did not care for the syntax. The participant answering on Robot Framework found that it made writing test cases easy. They also found Docker to be a good tool for software development, as containers made by others always work in the same way regardless of the environment. It had made moving things easier. One participant answering on Kaa noted that while Kaa was still incomplete, it worked surprisingly well. Other participant answering on Kaa noted that it is free to use with no payments and good for developing for fun. They suspected that in a corporation you have to use what they decide.

Based on question 29, IoT was seen as an interesting topic of the future. Even a participant, that had been sceptic beforehand, said that it is not just empty talk. IoT requires a sizeable stack of technologies; a single tool does not suffice for developing an IoT-solution. The IoT-field is constantly changing and the best tools for the job change often.

TABLE 6 below shows how many participants expected strengths and problems and how many experienced any for each area. Rest of the participants either did not expect any strengths or problems or did not experience any.

TABLE 6: Strengths and problems by area

Area	Strengths		Problems	
	Expected	Experienced	Expected	Experienced
APIs	6	6	6	8
Development tools	7	7	6	10
Information sources	9	7	6	9

6.4 Research question results

The research question was: "How do the expectations for Internet of Things service development user experience differ from the actual reality of developing them?" Secondary question was: "How do these expectations affect the user experience?"

Several different frameworks were used by the participants for developing their IoT service. Many of the frameworks used for this research were still under development and were found lacking in ways because of this. Expectations for the frameworks themselves were generally met with a mean of 6.2 on question 14 seen in TABLE 5, even when other questions for this area had lower means for their answers. This was the only area with such results, but other are not fully comparable due to this area having so many different Likert scale questions while other areas only have two.

APIs have a mean of 6,0 for expectations question 18 in TABLE 5 and a standard deviation of 0,943. It also had one participant to not agree with a 4. Framework they were answering for was Python.

Development tools was the area for which the participants were most pleased about. Expectations were met the most for this area with a mean of 6,2 and a standard deviation of 0,789 for question 23, as seen in TABLE 5. Six of the answers for expectations were 7s. This is the only area with over half of the answers being 7s. Rest of the answers were 6s with no 5s.

Information sources is the area which met the expectations worst with a mean of 5,4 for question 28 in TABLE 5. It had the highest standard deviation with 1,160, with the most spread out results. An outlier with the lowest answer for met expectations in any area was for JAX-RS with a 3. No other answer for expectations was this low. However, this area also had half of the answers be 5s, the highest amount out of all areas.

Based on TABLE 6, while number of participants expecting and experiencing strengths were the same for two areas out of three, more participants experienced problems in all areas. Nevertheless, while over half of all participants expected some problems and all experienced some with development tools, it was still the area where expectations were met the best based on question 23, as seen in TABLE 5. Many of the problems were reported to be inconsequential or isolated cases.

Information sources was the only area where there were less strengths experienced than expected. Two participants did not have their expectations met at all while other two had their expected strengths met only partially. It was also the area where expectations for the area were met the worst as seen in question 28.

It seems that having more problems than expected might not significantly affect the feeling of having expectations be met. While expected strengths not being met has more of an effect on expectations being met. Understandably, not experiencing problems that they might have been expecting is not seen as a detriment, but as a positive. While not having their expectations on strengths be met is seen as a failure. The question for meeting expectations for each area is most likely seen through positive for the participants even if the question does not specify this. So, if negative expectations are not met, this does not likely reduce the level of agreement. Meanwhile, failing to meet positive expectations is expected to reduce the level of agreement. With expectations being a part of user experience, as seen in chapter 3.6, expectations being met or not affects user experience. Assuming that the questions on each area meeting expectations is seen through positive, meeting them can be seen as good user experience.

Strengths could possibly be seen as pragmatic qualities mentioned in chapter 3.4. With information sources missing some pragmatic qualities in the form of unmet expected strengths, problems that participants experienced had a greater effect on their evaluation of the area, and thus user experience, than it otherwise would have.

6.5 Discussion on the form and procedure

The questions on how the frameworks met expectations in different areas was phrased in a way that left some ambiguity in answering. This was mentioned by a participant while filling the questionnaire. An additional field for explaining the answer would have been useful to reduce this ambiguity.

The researcher had also spent some time observing and interacting with the participants. This made it possible for the participants to be able to give their comments more freely while filling the questionnaire, as the person next to them was not a total stranger.

Ideally there would have been an interview before and during the course in addition to after. This opportunity to interview these students was presented when they were already well into the project, making earlier interviews impossible. Additionally, the modified form was not ready at first, requiring some feedback and development before starting the interviews. This development also took some time.

The questionnaire was disparate on the number of questions by topic area. This made comparing the answers with a Likert scale between the first topic area and the rest unequal, because the first topic area had so many more questions.

Question 4, their estimation and amount of time spent using the framework, gave similar results as the students had started using their respective frameworks for the particular assignment. One gave a much higher estimation, but this might vary due to the way they calculated it.

6.6 Limitations

The sample of participants was small and unrepresentative for a questionnaire study, even with the possibility of interview-like commenting given to the participants. A full interview might have given more information here.

A definite limitation is that the questionnaire was conducted only once, towards the end of the project. It would have been better to ask the questions about expectations before starting the project. Now the participants had to reflect on their expectations towards the end of the project, when they had already been using their respective frameworks almost to the full extent of the project. Answering this late might have distorted their responses somewhat from what they would have answered if the questions would have been asked before starting the project.

Because the participants could choose what they answer on, this created some problems at least in one case. Because one participant did not have a concrete framework that they were using, they answered on a whole programming language. Instead, maybe they could have concentrated on the modules they were using for help. In retrospective, this is something that could have been suggested for them to concentrate on when discussing what to answer upon.

Limitation is that for each individual framework, there was only one or two participants answering on it, yet there were many very different frameworks. As the questionnaire was conducted with a group of people working on a single project, there were participants working on different parts of the software project. Therefore, there were usually only one or two people working with any one framework. As such, the answers were on many different frameworks that cannot be compared to one another. A possibility on how to use this as an advantage could have been to take the project as a whole instead of each participant as only an individual. For example, by having them answer questions together as a group on how the different parts of the project come together as a complete whole.

7 CONCLUSION

This thesis explored the topic of user experience of developing Internet of Things services from the viewpoint of programmers' expectations. The research question was: "How do the expectations for Internet of Things service development user experience differ from the actual reality of developing them?" Secondary question was: "How do these expectations affect the user experience?"

These were approached by first going over the topic of Internet of Things in chapter 2. First, the concept of IoT was explored on a higher level, followed by a look into different kinds of software architectures and designs for IoT. Then some example applications were listed, after which challenges and limitations of IoT were examined. Last, energy efficiency and security of IoT were expanded upon. In chapter 3, user experience was inspected on a general level. First user experience was compared to usability, after which user experience was reviewed on different definitions, the hedonic and pragmatic sides of user experience, components of user experience, anticipation and expectations, on how to evaluate user experience, shortly on critique and limitations of user experience and a look into programmers as users. Chapter 4 reviewed designing user experience of Internet of Things and design principles and guidelines for IoT systems. Chapter 5 explained the research methods of the empirical study. First the steps of developing the questionnaire were reviewed. This was followed by chapters on recruiting participants and on ethical considerations. Chapter 6 went over and discussed the results of the questionnaires from different angles. Also, the questionnaire form and procedure and limitations of the research were explored.

The participants for this research were all students with little previous experience in development and who had just started working with IoT some weeks earlier. For frameworks and tools, all participants at least somewhat agreed that these areas of development met their expectations. APIs had a case where the participant did not agree or disagree while information sources had a participant slightly disagree.

With regards to the research questions, in addition to a general overview, three different areas of development user experience were reviewed: APIs, development tools and information sources. For each area, different number of participants expected different strengths and problems. For APIs and development tools, the number of participants that expected and experienced strengths was the same, while information sources had less participants with strengths experienced than expected. This was the area for which the expectations were met the worst. For problems, each area had more participants who experienced than expected problems. Over half of the participants had expected specific problems for development tools, but all the participants had experienced some problems. Yet, this was the area for which the expectations were met the best.

Positively surprising the programmers through unfulfilled negative expectations seems to have less of an effect on overall evaluation of expectations and user experience than not meeting expectations for strengths. Strengths could possibly be seen as pragmatic qualities of user experience. With information sources missing some pragmatic qualities, problems that participants experienced had a greater negative effect on their evaluation and user experience of the area than it otherwise would have. For the design and development process this means that it is perhaps better to gear programmers towards being prepared for setbacks or less sensational outcomes, than it is to instill highly positive expectations before a development process is fulfilled.

Further research on the topic could explore try to conduct similar empirical research, but with several participants working with the same framework. This would give a more in depth look into each framework, with a single participant not being the only source of data for each framework. Another possibility is to conduct similar research with either more experienced participants or with frameworks that are more mature for comparison.

REFERENCES

Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., & Ayyash, M. (2015). Internet of things: A survey on enabling technologies, protocols, and applications. *Communications Surveys & Tutorials, IEEE, 17*(4), 2347-2376.

Alves, R., Valente, P., & Nunes, N. J. (2014). The state of user experience evaluation practice. *Proceedings of the 8th Nordic Conference on Human-Computer Interaction: Fun, Fast, Foundational,* 93-102.

Ashton, K. (2009). That 'internet of things' thing. *RFiD Journal, 22*(7), 97-114.

Atzori, L., Iera, A., & Morabito, G. (2010). The internet of things: A survey. *Computer Networks, 54*(15), 2787-2805.

Bandyopadhyay, D., & Sen, J. (2011). Internet of things: Applications and challenges in technology and standardization. *Wireless Personal Communications, 58*(1), 49-69.

Bargas-Avila, J. A., & Hornbæk, K. (2011). Old wine in new bottles or novel challenges: A critical analysis of empirical studies of user experience. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems,* 2689-2698.

Bevan, N. (2009). What is the difference between the purpose of usability and user experience evaluation methods. *Proceedings of the Workshop UXEM, , 9*

Chaqfeh, M. A., & Mohamed, N. (2012). Challenges in middleware solutions for the internet of things. *Collaboration Technologies and Systems (CTS), 2012 International Conference On,* 21-26.

Chen, Y. (2012). Challenges and opportunities of internet of things. *Design Automation Conference (ASP-DAC), 2012 17th Asia and South Pacific,* 383-388.

Coetzee, L., & Eksteen, J. (2011). The internet of things-promise for the future? an introduction. *IST-Africa Conference Proceedings, 2011,* 1-9.

Covington, M. J., & Carskadden, R. (2013). Threat implications of the internet of things. *Cyber Conflict (CyCon), 2013 5th International Conference On,* 1-12.

Da Xu, L., He, W., & Li, S. (2014). Internet of things in industries: A survey. *Industrial Informatics, IEEE Transactions On, 10*(4), 2233-2243.

den Ouden, E., Yuan, L., Sonnemans, P. J., & Brombacher, A. C. (2006). Quality and reliability problems from a consumer's perspective: An increasing problem overlooked by businesses? *Quality and Reliability Engineering International, 22*(7), 821-838.

Diefenbach, S., & Hassenzahl, M. (2011). The dilemma of the hedonic– Appreciated, but hard to justify. *Interacting with Computers, 23*(5), 461-472.

Elkhodr, M., Shahrestani, S., & Cheung, H. (2013). The internet of things: Vision & challenges. *TENCON Spring Conference, 2013 IEEE,* 218-222.

Evans, D. (2011). The internet of things: How the next evolution of the internet is changing everything. *CISCO White Paper, 1,* 1-11.

Forlizzi, J., & Battarbee, K. (2004). Understanding experience in interactive systems. *Proceedings of the 5th Conference on Designing Interactive Systems: Processes, Practices, Methods, and Techniques,* 261-268.

Forlizzi, J., & Ford, S. (2000). The building blocks of experience: An early framework for interaction designers. *Proceedings of the 3rd Conference on Designing Interactive Systems: Processes, Practices, Methods, and Techniques,* 419-423.

Fronemann, N., & Peissner, M. (2014). User experience concept exploration: User needs as a source for innovation. *Proceedings of the 8th Nordic Conference on Human-Computer Interaction: Fun, Fast, Foundational,* 727-736.

Gamundani, A. M. (2015). An impact review on internet of things attacks. *Emerging Trends in Networks and Computer Communications (ETNCC), 2015 International Conference On,* 114-118.

Gollakota, S., Reynolds, M. S., Smith, J. R., & Wetherall, D. J. (2014). The emergence of RF-powered computing. *Computer, 47*(1), 32-39.

Gould, J. D., & Lewis, C. (1985). Designing for usability: Key principles and what designers think. *Communications of the ACM, 28*(3), 300-311.

Guinard, D., Trifa, V., Karnouskos, S., Spiess, P., & Savio, D. (2010). Interacting with the soa-based internet of things: Discovery, query, selection, and on-demand provisioning of web services. *Services Computing, IEEE Transactions On, 3*(3), 223-235.

Hachem, S., Teixeira, T., & Issarny, V. (2011). Ontologies for the internet of things. *Proceedings of the 8th Middleware Doctoral Symposium,* 3.

Hassenzahl, M. (2008). User experience (UX): Towards an experiential perspective on product quality. *Proceedings of the 20th International Conference of the Association Francophone D'Interaction Homme-Machine,* 11-15.

Hassenzahl, M., Diefenbach, S., & Göritz, A. (2010). Needs, affect, and interactive products–Facets of user experience. *Interacting with Computers, 22*(5), 353-362.

Hassenzahl, M., Platz, A., Burmester, M., & Lehner, K. (2000). Hedonic and ergonomic quality aspects determine a software's appeal. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems,* 201-208.

Hassenzahl, M., & Tractinsky, N. (2006). User experience-a research agenda. *Behaviour & Information Technology, 25*(2), 91-97.

Hu, F., Xie, D., & Shen, S. (2013). On the application of the internet of things in the field of medical and health care. *Green Computing and Communications (GreenCom), 2013 IEEE and Internet of Things (iThings/CPSCom), IEEE International Conference on and IEEE Cyber, Physical and Social Computing,* 2053-2058.

Huang, Y., & Li, G. (2010). Descriptive models for internet of things. *Intelligent Control and Information Processing (ICICIP), 2010 International Conference On,* 483-486.

Hurlburt, G. F., Voas, J., & Miller, K. W. (2012). The internet of things: A reality check. *IT Professional,* (3), 56-59.

Ibargoyen, A., Szostak, D., & Bojic, M. (2013). The elephant in the conference room: Let's talk about experience terminology. *CHI'13 Extended Abstracts on Human Factors in Computing Systems,* 2079-2088.

International Organization for Standardization. (2010). ISO 9241-210:2010(en) Ergonomics of human-system interaction — Part 210: Human-centred design for interactive systems. Retrieved from https://www.iso.org/obp/ui/#iso:std:iso:9241:-210:ed-1:v1:en

Jayakumar, H., Lee, K., Lee, W. S., Raha, A., Kim, Y., & Raghunathan, V. (2014). Powering the internet of things. *Proceedings of the 2014 International Symposium on Low Power Electronics and Design,* 375-380.

Kanuparthi, A., Karri, R., & Addepalli, S. (2013). Hardware and embedded security in the context of internet of things. *Proceedings of the 2013 ACM Workshop on Security, Privacy & Dependability for Cyber Vehicles,* 61-64.

Kardeby, V., Jennehag, U., & Gidlund, M. (2015). Power consumption for global information dissemination in the internet of things. *Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 2015 IEEE Tenth International Conference On,* 1-6.

Khan, R., Khan, S. U., Zaheer, R., & Khan, S. (2012). Future internet: The internet of things architecture, possible applications and key challenges. *Frontiers of Information Technology (FIT), 2012 10th International Conference On,* 257-260.

Kopetz, H. (2011). Internet of things. *Real-time systems* (pp. 307-323) Springer.

Kortuem, G., Kawsar, F., Fitton, D., & Sundramoorthy, V. (2010). Smart objects as building blocks for the internet of things. *Internet Computing, IEEE, 14*(1), 44-51.

Lallemand, C., Koenig, V., & Gronier, G. (2014). How relevant is an expert evaluation of user experience based on a psychological needs-driven approach? *Proceedings of the 8th Nordic Conference on Human-Computer Interaction: Fun, Fast, Foundational,* 11-20.

Law, E. L. (2011). The measurability and predictability of user experience. *Proceedings of the 3rd ACM SIGCHI Symposium on Engineering Interactive Computing Systems,* 1-10.

Law, E. L., Roto, V., Hassenzahl, M., Vermeeren, A. P., & Kort, J. (2009). Understanding, scoping and defining user experience: A survey approach. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems,* 719-728.

Law, E. L., Roto, V., Vermeeren, A. P., Kort, J., & Hassenzahl, M. (2008). Towards a shared definition of user experience. *CHI'08 Extended Abstracts on Human Factors in Computing Systems,* 2395-2398.

Lazar, J., Jones, A., & Shneiderman, B. (2006). Workplace user frustration with computers: An exploratory investigation of the causes and severity. *Behaviour & Information Technology, 25*(03), 239-251.

Lee, S. W., Prenzel, O., & Bien, Z. (2013). Applying human learning principles to user-centered IoT systems. *Computer,* (2), 46-52.

Liu, V., Parks, A., Talla, V., Gollakota, S., Wetherall, D., & Smith, J. R. (2013). Ambient backscatter: Wireless communication out of thin air. *ACM SIGCOMM Computer Communication Review, 43*(4), 39-50.

Matharu, G. S., Upadhyay, P., & Chaudhary, L. (2014). The internet of things: Challenges & security issues. *Emerging Technologies (ICET), 2014 International Conference On,* 54-59.

McCarthy, J., & Wright, P. (2004). Technology as experience. *Interactions, 11*(5), 42-43.

Meng, Q., & Jin, J. (2011). The terminal design of the energy self-sufficiency internet of things. *Control, Automation and Systems Engineering (CASE), 2011 International Conference On,* 1-5.

Miorandi, D., Sicari, S., De Pellegrini, F., & Chlamtac, I. (2012). Internet of things: Vision, applications and research challenges. *Ad Hoc Networks, 10*(7), 1497-1516.

Miraz, M. H., Ali, M., Excell, P. S., & Picking, R. (2015). A review on internet of things (IoT), internet of everything (IoE) and internet of nano things (IoNT). *Internet Technologies and Applications (ITA), 2015,* 219-224.

Myllärniemi, V., Kujala, S., Raatikainen, M., & Sevón, P. (2016). *Development as a journey: Factors supporting adoption and use of software frameworks.* Unpublished manuscript.

Nielsen, J. (1993). *Usability engineering* AP PROFESSIONAL.

Nieminen, J., Savolainen, T., Isomäki, M., Patil, B., Shelby, Z., & Gomez, C. (2015). *IPv6 over BLUETOOTH(R) low energy [PROPOSED STANDARD]*

Olsson, T. (2014). Layers of user expectations of future technologies: An early framework. *CHI'14 Extended Abstracts on Human Factors in Computing Systems,* 1957-1962.

Palattella, M. R., Accettura, N., Vilajosana, X., Watteyne, T., Grieco, L. A., Boggia, G., & Dohler, M. (2013). Standardized protocol stack for the internet of (important) things. *Communications Surveys & Tutorials, IEEE, 15*(3), 1389-1406.

Parks, A. N., Liu, A., Gollakota, S., & Smith, J. R. (2014). Turbocharging ambient backscatter communication. *ACM SIGCOMM Computer Communication Review, , 44*(4) 619-630.

Prabhakararao, S., Cook, C., Ruthruff, J., Creswick, E., Main, M., Durham, M., & Burnett, M. (2003). Strategies and behaviors of end-user programmers with interactive fault localization. *Human Centric Computing Languages and Environments, 2003. Proceedings. 2003 IEEE Symposium On,* 15-22.

Roto, V. (2007). User experience from product creation perspective. *Towards a UX Manifesto Workshop,* 31-34.

Roto, V., Law, E. L., Vermeeren, A., & Hoonhout, J. (2011). User experience white paper. *Bringing Clarity to the Concept of User Experience,*

Rousi, R. (2014). Unremarkable experiences–designing the user experience of elevators. *Swedish Design Research Journal, 1*(14), 57-64.

Rowland, C., Goodman, E., Charlier, M., Light, A., & Lui, A. (2015). *Designing connected products: UX for the consumer internet of things* " O'Reilly Media, Inc.".

Saeed, A., Ammar, M., Harras, K. A., & Zegura, E. (2015). Vision: The case for symbiosis in the internet of things. *Proceedings of the 6th International Workshop on Mobile Cloud Computing and Services,* 23-27.

Salman, Y., Abu-Issa, A., Tumar, I., & Hassouneh, Y. (2015). A proactive multi-type context-aware recommender system in the environment of internet of things. *Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), 2015 IEEE International Conference On,* 351-355.

Scaffidi, C., Shaw, M., & Myers, B. (2005). Estimating the numbers of end users and end user programmers. *Visual Languages and Human-Centric Computing, 2005 IEEE Symposium On,* 207-214.

Siekkinen, M., Hiienkari, M., Nurminen, J. K., & Nieminen, J. (2012). How low energy is bluetooth low energy? comparative measurements with zigbee/802.15. 4. *Wireless Communications and Networking Conference Workshops (WCNCW), 2012 IEEE,* 232-237.

Tan, L., & Wang, N. (2010). Future internet: The internet of things. *Advanced Computer Theory and Engineering (ICACTE), 2010 3rd International Conference On, , 5* V5-376-V5-380.

Tokkonen, H., & Saariluoma, P. (2013). How user experience is understood? *Science and Information Conference (SAI), 2013,* 791-795.

Väänänen-Vainio-Mattila, K., Roto, V., & Hassenzahl, M. (2008). Towards practical user experience evaluation methods. *EL-C.Law, N.Bevan, G.Christou, M.Springett & M.Lárusdóttir (Eds.) Meaningful Measures: Valid Useful User Experience Measurement (VUUM), ,* 19-22.

Varsaluoma, J., & Sahar, F. (2014). Usefulness of long-term user experience evaluation to product development: Practitioners' views from three case studies. *Proceedings of the 8th Nordic Conference on Human-Computer Interaction: Fun, Fast, Foundational,* 79-88.

Vermeeren, A. P., Law, E. L., Roto, V., Obrist, M., Hoonhout, J., & Väänänen-Vainio-Mattila, K. (2010). User experience evaluation methods: Current state and development needs. *Proceedings of the 6th Nordic Conference on Human-Computer Interaction: Extending Boundaries,* 521-530.

Wallach, D., & Scholz, S. C. (2012). User-centered design: Why and how to put users first in software development. *Software for people* (pp. 11-38) Springer.

Welbourne, E., Battle, L., Cole, G., Gould, K., Rector, K., Raymer, S., . . . Borriello, G. (2009). Building the internet of things using RFID: The RFID ecosystem experience. *Internet Computing, IEEE, 13*(3), 48-55.

Yang, Z., Peng, Y., Yue, Y., Wang, X., Yang, Y., & Liu, W. (2011). Study and application on the architecture and key technologies for IOT. *Multimedia Technology (ICMT), 2011 International Conference On,* 747-751.

Yogasara, T., Popovic, V., Kraal, B. J., & Chamorro-Koc, M. (2011). General characteristics of anticipated user experience (AUX) with interactive products. *Proceedings of IASDR2011: The 4th World Conference on Design Research: Diversity and Unity,* 1-11.

APPENDIX 1 CONSENT FORM

Tiedote tutkittaville ja suostumus tutkimukseen osallistumisesta

Tutkija:
Niko Mehtonen
S-posti:
Puh:

Tutkimuksen Ohjaaja:
FT Rebekah Rousi
S-posti:
Puh:

Tutkimuslaitos: Jyväskylän Yliopisto, Suomi

Tutkimuksesta:
Tämä on kysely Esineiden Internet eli Internet of Things kehittämisestä. Tarkoituksena on selvittää, kuinka kehityksessä käytettävät ohjelmistokehykset vastaavat niihin liittyviä odotuksia kehitystyössä.

Menettely:
Tutkimus suoritetaan kyselynä. Kerään ensin taustatietoja ennen kyselyä. Kyselyyn sisältyy sekä avoimia kysymyksiä, että kysymyksiä, joissa valitaan, kuinka samaa mieltä osallistuja on väitteen kanssa. Lisäksi istunnot äänitallennetaan asiaan liittyvien kommenttien ja huomioiden keräämiseksi.

Luottamuksellisuus, nimettömyys and turvallisuus:
Vastaukset käsitellään luottamuksellisesti, nimettömästi ja mahdollisimman turvallisesti. Taustatiedoissa kerättyä nimeä käytetään vain mahdollisiin vastausten selvennyksiin, tai jos osallistuja itse haluaa päästä tarkistamaan omat vastauksensa.

Tutkimusdatan säilytys:

Tämän tutkimuksen tuloksia käytetään pro gradu tutkimukseen ja ne tuhotaan muun tutkimudatan ohella, kun tutkimus on valmis.

Tutkittavien oikeudet:
Osallistuminen tutkimukseen on täysin vapaaehtoista. Osallistujilla on oikeus keskeyttää tutkimus ilman mitään seurauksia. Järjestelyt ja raportointi ovat luottamuksellisia. Voit halutessasi päästä tarkistamaan vastauksesi ennen kuin niitä käytetään.

Tutkittavan suostumus tutkimukseen osallistumiseen:

1. **Olen tutustunut tutkimuksen tarkoitukseen ja sisältöön.**

2. **Ymmärrän, että minulle ei aiheudu vahinkoa tutkimukseen osallistumisesta.**

3. **Suostun osallistumaan mittauksiin ja menettelyihin ohjeiden mukaan.**

4. **Voin peruuttaa tai lopettaa osallistumisen, tai kieltäytyä osallistumasta mittauksiin missä tahansa vaiheessa.**

5. **Tutkimuksen tuloksia voidaan käyttää tieteelliseen raportointiin (esim. julkaisuihin) tavalla, joka estää yksittäisen osallistujan tunnistamisen.**

6. **Annan luvan istunnon äänitallentamiseen (ympyröi vaihtoehto):**

   Kyllä / Ei

____

Päiväys Osallistujan allekirjoitus

____

Päiväys Tutkijan allekirjoitus

APPENDIX 2 QUESTIONNAIRE FORM

ESINEIDEN INTERNET-TUTKIMUS

Vastaa kysymyksiin parhaasi mukaan. Väittämissä vastaukset ovat väliltä 1=Täysin eri mieltä, 7=Täysin samaa mieltä.
Vastaa väittämiin korostamalla (maalaa ja sitten Ctrl+B) haluamasi numero.

Valittu ohjelmistokehys:

Software framework. Alusta, joka tarjoaa tietyn toiminnallisuuden toteuttamista helpottavia, valmiita ratkaisuja.

Kysymys	Vastaus
1. Kuvaile sovelluskehyksen tarkoitusta Esineiden Internet (Internet of Things, IoT) käytössä.	
2. Kokemuksesi vastaavista teknologioista.	
3. Arvioi, kuinka monta tuntia **odotit** kuluvan sovelluskehyksen asentamiseen tai asentamiseen liittyvän tiedon etsimiseen ja kuinka paljon siihen oikeasti käytit.	
4. Arvioi, kuinka monta tuntia olet käyttänyt sovelluskehystä.	

5. Sovelluskehys tuntuu hyvältä.	1	2	3	4	5	6	7
6. Nautin sovelluskehyksen käyttämisestä.	1	2	3	4	5	6	7
7. Sovelluskehyksen käyttäminen on palkitsevaa.	1	2	3	4	5	6	7
8. Jos haluat, kerro lisää kokemuksistasi liittyen kysymyksiin 5-7.							
9. Sovelluskehys täyttää vaatimukseni.	1	2	3	4	5	6	7
10. Sovelluskehyksen käyttäminen on turhauttavaa.	1	2	3	4	5	6	7
11. Sovelluskehys on helppokäyttöinen.	1	2	3	4	5	6	7
12. Joudun käyttämään liikaa aikaa asioiden korjaamiseen käyttäessäni sovelluskehystä.	1	2	3	4	5	6	7
13. Jos haluat, kerro lisää kokemuksistasi liittyen kysymyksiin 9-12.							
14. Sovelluskehys vastaa odotuksiani.	1	2	3	4	5	6	7

Ohjelmointirajapinnat
Application programming interface, API. Sovelluskehyksen rajapinnat, ohjaimet ja tietotyypit.

15. Ohjelmointirajapinnat tukevat tehtävän toteuttamista.	1	2	3	4	5	6	7
16. Kuvaile, millaisia ongelmia **odotit** ohjelmointirajapinnoilta, ja mitä ongelmia sinulla on lopulta ollut.							

17. Kuvaile, millaisia vahvuuksia **odotit** ohjelmointirajapinnoilta, ja mitä vahvuuksia niillä on lopulta ollut.							
18. Ohjelmointirajapinnat vastaavat odotuksiani.	1	2	3	4	5	6	7

Kehitystyökalut
Development tools. Sovellukset, joilla tuotat ja käytät koodia: kehitysympäristöt, tekstieditorit, kääntäjät, emulaattorit jne.

19. Mitä kehitystyökaluja olet käyttänyt kyseisen sovelluskehyksen kanssa työskentelemiseen?							
20. Kehitystyökalut tukevat tehtävän toteuttamista.	1	2	3	4	5	6	7
21. Kuvaile, millaisia ongelmia **odotit** kehitystyökaluilta, ja mitä ongelmia sinulla on lopulta ollut.							
22. Kuvaile, millaisia vahvuuksia **odotit** kehitystyökaluilta, ja mitä vahvuuksia niillä on lopulta ollut.							
23. Kehitystyökalut vastaavat odotuksiani.	1	2	3	4	5	6	7

Tietolähteet
Esim. dokumentaatio, keskustelupalstat, koodiesimerkit.

24. Mitä tietolähteitä **odotit** käyttäväsi sovelluskehykseen liittyen, ja mitä tietolähteitä olet käyttänyt.							
25. Tiedonlähteet tukevat sovelluskehyksen käyttämistä.	1	2	3	4	5	6	7
26. Kuvaile, millaisia ongelmia **odotit** tietolähteiltä, ja mitä ongelmia sinulla on lopulta ollut.							
27. Kuvaile, millaisia vahvuuksia **odotit** tietolähteiltä, ja mitä vahvuuksia niillä on lopulta ollut.							
28. Tietolähteet vastaavat odotuksiani.	1	2	3	4	5	6	7

Muuta

29. Omat kommentit liittyen aiheeseen. Mitä vain ajatuksia tulee mieleen.	

APPENDIX 3 VERBAL INSTRUCTIONS

Tämä on tutkimus liittyen Esineiden Internet eli Internet of Things kehittämiseen. Tutkimuksen näkökulmana on odotukset, joten haluaisin, että kyselyä täyttäessäsi miettisit ja vastaisit pyydettäessä odotuksien kautta.

Taustatiedoissa kysytään nimeäsi. Sitä ei käytetä yksittäisten osallistujien tunnistamiseen, vaan se on mahdollisia vastausten selvennyksiä varten. Voit myös päästä tarkistamaan omat vastauksesi halutessasi.

Tiedostot ja monisteet on numeroitu. Tämä on vain vastausten keräämistä ja tiedon tallentamista varten.

Kyselytilanne äänitetään asiaan liittyvien kommenttien keräämiseksi. Voit kieltäytyä tästä halutessasi.

Jos sinulla on tai tulee kysymyksiä tai kommentteja, niin kysy vapaasti.

Kiitos avustasi!

APPENDIX 4 BACKGROUD INFORMATION FORM

TAUSTATIEDOT

Nimi:
Päivämäärä:
Ikä:
Sukupuoli: Mies / Nainen
Ammatti:
Ala:
Koulutustausta (korkein koulutusaste):
Teknologiset taidot (alleviivaa sopiva): Perusteet / Aloittelija / Keskitaso / Edistynyt / Asiantuntija
Kokemus alan uusista teknologioista. Voit myös mainita esimerkkejä: