Experiences in Applying Service Design to Digital Services

Stefanie Hofemann[1], Mikko Raatikainen[1], Varvana Myllärniemi[1], and Terho Norja[2]

[1]Aalto University, Finland
{firstname.lastname}@aalto.fi
[2]Steeri, Finland
terho.norja@steeri.fi

Abstract. An increasing number of services is mainly provided through digital channels and thus, implemented as software. Nevertheless, many companies struggle with developing digital services that are considered valuable by the users. Recently, service design has emerged as an approach to design better customer experience for services. We describe our experiences with a service design approach, and specifically prototyping, to explore user needs for a digital meeting scheduling service (MSS). We created an interactive prototype and paper prototypes and used them in a prototype test session with potential users to explore different design alternatives. The experiences include the peculiarities of service design for digital services as well as challenges in prototyping. The results indicate service design as a promising approach to develop digital services that better meet user needs. However, challenges exist on a practical level, such as operationalizing the value-in-use concept, applying service design for digital services, and lack of practical guidelines for prototyping.

Keywords: service design, prototyping, service-dominant logic, digital service.

1 Introduction

In today's fast changing economy, it has become increasingly important to develop software that meets users' and other stakeholders' needs. However, the development of software is often still technology-driven. This can lead to technically superior solutions that are not necessarily considered valuable by the customers [1]. In recent years, service design (SD) has evolved as a new discipline, and it is often described as the discipline that brings design thinking and designer's methods into services [2]. Design thinking has been increasingly acknowledged as beneficial for innovation and developing solutions to customers' problems [3]. Design thinking is characterized by first focusing on identifying the problem and exploring possible solutions; only after that on how to implement these solutions, instead of restricting one's thinking by implementation constraints in the beginning [4].

The most commonly used service design methods are prototyping and visualizations [5]. Prototypes have been used in various disciplines, but the understanding of what they

A. Jedlitschka et al. (Eds.): PROFES 2014, LNCS 8892, pp. 134–148, 2014.

are varies among them. While in software development prototypes are typically seen as a simplified version of the final software, in SD, most visualizations and other artifacts can be considered a prototype [6].

In the field of service design, few studies have focused on services that are mainly distributed through digital channels. Instead, most publications in service design literature focus on case examples from traditional service industries, such as airlines, restaurant, and public services.

This paper studies how to apply service design in general and prototyping in particular to the development of digital services to gain better understanding of users' needs. The study was carried out with an industrial partner to the case of a meeting scheduling service (MSS). Thus, we aim to investigate two key questions:

RQ1 How does the development of digital services benefit from service design?
RQ2 What are the challenges in applying service design to the development of digital services?

The remainder of this paper is organized as follows: Section 2 presents previous work on services, service design, and prototypes. Section 3 introduces the industry case. Section 4 describes the research method. Section 5 presents the results. Section 6 discusses the findings and Section 7 draws conclusions.

2 Previous Work

This section describes the paradigm shift to service dominant logic, followed by the concept of service design, and ends with providing an overview of prototyping.

2.1 Services: The Paradigm Shift to Service-Dominant Logic

Services have often been defined in relation to goods and described based on characteristics that differentiate them from goods. The most commonly cited characteristics are intangibility, heterogeneity, inseparability and perishability, also known as IHIP-characteristics [7]. In this goods-dominant logic (G-D logic), services are considered inferior to goods [8]. However, in current service management literature, the leading school of thoughts is service-dominant logic (S-D logic) [9], in which goods are merely considered as mechanisms for the distribution of services [10] Thus, a service offering might include tangible and intangible elements. This paradigm shift entails a turn in the view on value creation. One definition of value is as the trade-off between benefits and sacrifices [11]. In G-D logic, value is embedded in the goods and referred to as *value-in-exchange* [12]. In S-D logic, value is referred to as *value-in-use* and means the value as perceived by the customer, which arises and changes over time [10].

Software is challenging to categorize as a product or service based on the IHIP-characteristics. While software is intangible, the other three characteristics of service, heterogeneity, inseparability, and perishability, only apply partially [8], [13]. Degree of customization [14] and revenue models [15] are common approaches to categorize

software as either product or service. Recently, Software-as-a-Service (SaaS) and cloud-based services have become popular business models. However, these models mainly refer to a change in the revenue model rather than a change in understanding of value as in S-D logic.

Digital services, such as online banking, have replaced some traditional services and new businesses have emerged, whose core offerings are digital [8]. In traditional services, the role of the front stage employees and their interaction with the customers is crucial for the service experience; in contrast, users of digital services might never get into personal contact with the service provider [16]. Moreover, many digital services, such as online social networks and online marketplaces, provide a platform for social interaction between their users [13]. The service experience of these services depends significantly on the behavior of other users instead of on the behavior of the front stage employees [17].

2.2 Service Design

Service Design originates in times, when services were defined based on the IHIP-characteristics. It was argued that not only products, but also services need design. The strong use of different designer's methods throughout the development process have been defined as the distinguishing characteristics of service design from other approaches to service development [18], [19]. The most common service design methods are prototyping and visualizations [5]. While most visualizations can be used as prototypes [6], not all prototypes are visualizations; for example, experience prototyping [20] and other enacting methods. Stickdorn [21] suggests five principles, which should guide the service design process: *user-centered*, *co-creative*, *sequencing*, *evidencing*, and *holistic*. Instead of user-centered, human-centered has also been suggested as one principle, in order to emphasize the inclusion of other stakeholders [22]. Co-creative refers to the active involvement of users and other stakeholders in the design process. Sequencing emphasizes the need to consider the whole customer journey. Evidencing refers to making the back stage process of the service visible to the customers. Holistic refers to considering also the context of use and thus, extending the principle of sequencing. Typically, a service design process is highly iterative and at each stage, it might be necessary to return to one of the previous stages [23].

Service design still seems to be dominated by the view that a service is different from a product rather than a higher-level concept, as in S-D logic [9]; however, it is seldom made explicit. Nevertheless, some authors have discussed the relation between service design and S-D logic [24]–[26]. Most principles of service design and S-D logic are overlapping and thus, service design is one approach to put the theoretic principles behind the S-D logic into practice [26]. Some authors refer to designing services driven by S-D logic as *design for service* instead of service design in order to make a clear distinction [13], [24]. However, there are different viewpoints concerning the relation of design for service and service design [24], [26]. One viewpoint is to consider design for service as the next step in the evolution of service design (Fig. 1). In this viewpoint product thinking equals G-D logic and service thinking equals S-D logic.

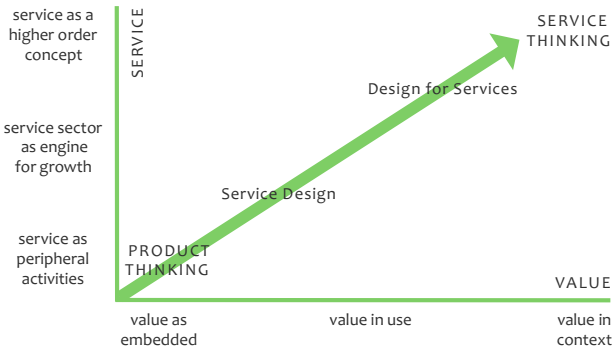

Fig. 1. The evolution from service design to design for service (Source: [25, p. 98])

Similarities exist between service design and other user-centered design disciplines, such as user-experience design. However, service design expands the focus to the long-term usage and across various channels [27].

2.3 Prototypes

Prototypes have been used in a variety of different disciplines; however, the purposes vary among different disciplines [6]. One way to refer to prototypes is as a "representation of a design idea" [28] and prototyping as "the activity of creating prototypes, or activities made possible by or with the prototype" [29]. In software development, and specifically user interfaces, prototyping has long been identified as one activity [30] that is performed before the final implementation [6] in order to evaluate hypotheses concerning the software to be build [31]. Recently, several incremental or iterative methods, such as agile software development, have evolved, in which the intermediate results can be considered a prototype representing a simplified, but almost ready version of the final system [32]. Technical prototypes are commonly used in software development to validate the technical feasibility of a solution; however, this is only one aspect of a whole solution – other aspects are role as well as look and feel [28]. Furthermore, prototypes have been used to evaluate the usability of a software.

In service design, prototypes are described as a learning tools [32], which can be used for various purposes with different levels of fidelity and at any stage of the process [28], [31] and thus, in a broader manner than traditionally in software development. Prototypes are not only used to evaluate a hypothesis or communication with different stakeholders [33], as typically in software development, but also for generation and exploration of ideas. In addition to prototypes that prototype different parts of the service, service prototypes can be used, which encompasses several service moments in order to prototype the holistic user experience [34].

Different frameworks exist to support prototyping [6], [31]. However, since there is not a single way to 'do it right' [33], the frameworks do not provide prototypes for

specific situations. Instead, they facilitate thinking about 'what' and 'how' to prototype. Consequently, prototyping in service design is a holistic approach or mind-set rather than merely a set of tools and activities [6].

3 Case Description: Meeting Scheduling Service (MSS)

The object of this study is a meeting scheduling system (MSS) for heterogeneous calendar systems. A software architecture and technical prototype for MSS (Fig. 2) have been developed in cooperation between Aalto University and the company Steeri, which is a service provider for Customer Relationship Management (CRM) solutions. MSS addresses the problem that current solutions for scheduling meetings mainly work effortlessly for persons within the same organization and using the same calendar system, such as Microsoft Exchange. Across organizational borders and between different calendar systems, no solution seems to exist to automatically check availabilities for easier meeting scheduling.

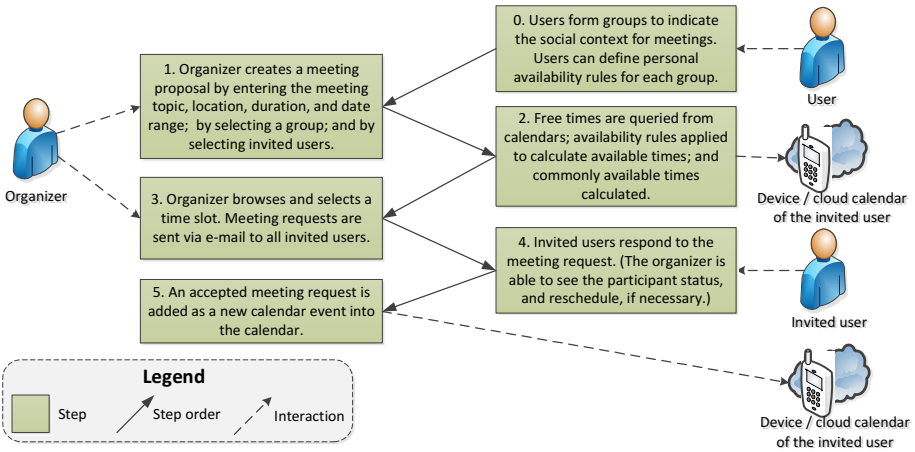

Fig. 2. MSS Scheduling Process

In contrast to existing solutions, MSS automatically retrieves free time slots from users' calendars and provides time slots that are free in the calendar of all meeting invitees to the meeting organizer as possible options for meeting times. Four basic assumptions were made regarding privacy concerns of the users and taken into consideration for the creation of the technical prototype and software architecture: First, users would not want to share free times with everybody. Thus, users first have to choose with whom they share their available time slots. Second, users will want to differentiate what times are shown as available based on so-called 'social context'. For each social context, e.g., a project team, users can set an availability rule to define what times are shown as available. For example, users can limit their availability for a certain project team to times in the afternoons. Third, users would not want

information other than free time slots to leave their calendar system. Fourth, meeting organizers should only see time slots that are free for all meeting invitees and not the time slots that are free for each individual invitee. The technical prototype was developed to address and focus on technical feasibility of the solution.

4 Research Method

The research design adheres to the explorative design science research approach [35]. The phases include gaining understanding about meeting scheduling context, creating the interactive and paper prototypes, and a prototype test session with potential users. Finally, the experiences are elaborated.

In order to gain initial understanding, we conducted a case study [36] consisting of a study of the existing technical prototype and existing material, and a half-day workshop with Steeri. The objective of the workshop was to gain better common understanding of the practices and tools to schedule meetings. The participants were the chief executive officer (CEO), the sales & marketing director, a senior consultant, and a software developer. In particular, the three first frequently interact with external parties, but only the CEO was beforehand familiar with technical prototype. In the workshop, a short overview and demo of the prototype were given, different kinds of meetings and the meeting scheduling process were elaborated, and challenges and solutions were gathered on post-it notes, prioritized and discussed. We audio-recorded the workshop and took field notes, including photographs.

After the workshop, an interactive prototype was created with the prototyping tool Axure (Fig. 3). In order to focus the feedback on the service concept, rather than details of the user interface, the interactive prototype had an unfinished look. The interactive prototype demonstrates the whole process, i.e., from taking the service into use to scheduling a meeting. The focus was on the aspects of the service relevant to the users instead of the technical implementation of the back-end. In addition to the interactive prototype, seven different paper prototypes (Fig. 3) were created to present different design alternatives in order to explore factors that the researchers considered the most critical from users' perspective: Two alternatives for the amount of information available to the organizer when selecting a time slot; two alternatives on how to set the availability rules; two alternatives showing different alternatives if no common free time slot was found; and one showing alternatives for taking location information into consideration in order to determine the available time slots more accurately.

In a two-hour session, the prototypes were discussed with potential users. We expected the following outcomes of the session: first, feedback for the service concept based on the interactive prototype and the design alternatives presented as paper prototypes; second, better understanding of users' needs for meetings scheduling and attitudes towards sharing of calendar data in general. The participants of the prototype test session frequently have to schedule meetings with people in various locations and across company borders. There were four participants in the session: three of them are part of the IT department (one manager, two specialists); the fourth participant is a manager in the marketing department.

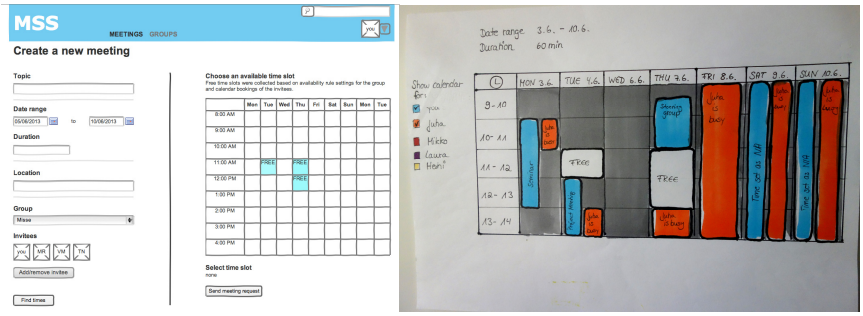

Fig. 3. Examples of the interactive prototype (left) and the paper prototype (right)

The session started by briefly introducing the goal of the session. Next, we showed the interactive prototype and asked the participants to evaluate the prototype. In order to spark discussion, different design alternatives were presented as paper prototypes. The researcher mostly asked questions to clarify certain statements and comment or to get feedback on specific topics. The session was audio recorded.

The data analysis started with extracting important points from the audio recordings and field notes from the initial workshop. Similarly after the prototype test sessions, we extracted the important statements and comments, resorting to the audio recording when necessary. The data analysis then interlaced with a re-analysis of the workshop data because we discovered differences between scheduling meeting behaviors, since different calendar access model were used in both companies. The later analysis focused on differences and similarities between the participants of the workshop and the participants of the prototype test session.

5 Results

The results include the observations and experiences from the workshop with Steeri as well as the prototype test session. This section describes the generalized findings based on the results.

5.1 Change in the Perception of Value

The perceived value varies from user to user and it can change over time, which implies that there is no value in a feature per-se. When applying the value-in-use concept, the user subjectively defines the value. There is rarely a 'one size fits all' solution in any complex service and thus, services need to support the individual customer journeys. For example, people at Steeri use an open calendar access model, i.e., they can see all calendar details of their colleagues, and they seem to use this information comprehensively for scheduling meetings. In contrast, the participants of the prototype test session, using a restricted calendar access model, mainly seemed to be interested in knowing free time slot in the calendar of their colleagues. Furthermore, there also seemed to be a difference depending on the position.

The specialists seemed to face more challenges that others do not prioritize the meetings that they schedule and thus, they seemed more interested in knowing more details concerning the schedule of others.

Value also depends on the sacrifices that a customer has to make. Surprisingly, for the participants of the prototype test session sharing information from their calendar did not constitute much privacy concerns. Rather, a reoccurring worry was the amount of meetings, and the risk that a service, such as MSS, could lead to having even more meetings. Consequently, the features focusing on privacy in the technical prototype cannot be considered generally valuable for all users.

5.2 Challenge to Consider the Whole Customer Journey

The focus of the technical prototype was on the functionality of scheduling meetings with people from different companies. Scheduling of internal meetings was left out of the scope. However, the results of the prototype test session indicate that people do not clearly distinguish between internal meetings and meetings with externals. Overall, they just wish to schedule meetings easily. Even though asked about scheduling meetings with people from other companies, in both, the workshop with Steeri and the prototype test session, a large amount of the discussion evolved around scheduling meetings with colleagues. Moreover, the participants of the prototype test session preferred not to have a separate service. While technical design sets borders clearly, these borders do not exist similarly in the users' mind. This can result in superior technical solution, but inadequate user experience, since the solution might not support the whole customer journey.

Similarly, the technical design focuses on features rather than the holistic customer journey. For example, easy adoption and how to connect with other users in order to share available time are crucial for the success of MSS; however, they had not been covered, when designing the technical prototype.

5.3 Applying Service Design to an Existing Technical Prototype

A SD process typically starts from the scratch to explore possibilities rather than from a technical prototype, as it was in the case of MSS. The technical prototype limited the exploration of different options and thus, the service concept is an incremental change rather than radically new compared to the existing technical prototype. The solution might have been different if the project had started with a service design approach to create the initial idea for the concept. However, discarding a technical prototype and software architecture denotes a significant change that is not necessarily wanted. This was also the case for MSS. Applying service design methods and principles, nevertheless, helped exploring and gaining deeper understanding of users' needs. Furthermore, the prototypes and visualizations facilitate better communication among the different stakeholders.

5.4 Service Design for Digital Services

Many service design methods focus on traditional services and thus, are not directly applicable when designing digital services. When designing traditional services, the interaction between the front stage employees and the customer is emphasized. Furthermore, they often take place in a specific physical space. In contrast in digital services, users interact with a software system. Moreover, the aim of many digital services, such as MSS, is to facilitate interaction between different users. This leads to less control of the service experience for the service provider, since the behavior of other users cannot be controlled in the same manner as the behavior of front stage employees. For example, the user experience of MSS depends significantly on how strict other users set the availability rules. Consequently, many characteristics of digital services are different from traditional services. Furthermore, some of the basic principles of service design do not apply in the same manner. For example, evidencing service takes a different form. Furthermore, the methods need consideration. For example, enacting techniques, which are common to prototype traditional services, would have not been suitable for MSS. Overall, SD provides little guidelines on methods for prototyping and implementing digital services.

5.5 Challenges in Choosing Prototyping Techniques

Due to the plethora of different techniques available for prototyping, it was challenging to choose suitable techniques for the given purpose. While there are some recommendations for which phase of the service design process some techniques are most suitable [37], [38], overall, the choice is left to the designer. While we were thinking prior to the prototype test session that it might be good to prototype the experience more holistically, we discovered during the session that prototyping only parts of MSS with the paper prototypes seems more suitable due to the early stage in the SD process and the focus on exploration. The holistic service experience can be prototyped at a later stage. In fact, it was challenging to achieve a service prototype for MSS: The technical prototype is only functioning on a specific device and thus, could not be easily used to retrieve actual data from the participants' calendars. Furthermore, it only covers parts of the service. The interactive prototype covers the whole process, but only simulates the service and does not retrieve actual calendar data. Thus, users could only imagine how it would work in practice, i.e., what kind of time suggestions they would get in real usage situations. However, as the participants are active users of electronic calendars, it seemed that they could imagine how the calendar data retrieval would work in practice.

The paper prototypes and the sketchy interactive prototype seemed to encourage open feedback, since they did not convey the notion of being close to the final version. We had a quick walkthrough of the interactive prototype, but then mainly focused on the paper prototypes, since they seemed to encourage more discussion than the interactive prototype. However, the chosen method did not seem to encourage proposing own ideas of the participants. The participants mostly focused on their preferences comparing the different design alternatives and possibly

proposing to combine them. However, they did not make own suggestions. In order to encourage generation of own ideas, other prototyping methods might have been more beneficial. The interactive prototype could be more beneficial in a later stage of the process, when the concept is more finalized and the focus is on evaluation rather than a more open exploration.

5.6 Nature of Prototypes

On the basis of our experience, it seems that it was beneficial to have different design alternatives. The alternatives reduced the likelihood of receiving purely affirmative feedback for a proposed solution, since they forced the participants to take a stand on what they like and what they do not like. For example, it was discovered to be more intuitive to set the availability rules based on the organizer of the meeting rather than social context. Another example was the preference of showing less information in order to select free time slots. As one participant commented: "I only want to see common free time slots. I don't care about what others have before or after". However, some participants also preferred having more information available.

Besides new ideas and selection between alternatives, prototypes could exclude certain features. However, this did not occur. For example, although the participants were discussing about the availability rules during the prototype test session, none of the participants mentioned why or how they would want to use availability rules. This might indicate that the availability rules would not be used much. From service design perspective, the findings suggest that users might not exclude superfluous features if they do not disturb them. This can result in unnecessarily rich and complex services.

6 Discussion

This section discusses the findings related to the research problems, namely digital service design and prototyping.

6.1 Digital Service Design

It requires a change in the mindset to consider software as a service rather than as a product. In particular, it is more than a change in business or delivery model, as in the case of changing to SaaS. Essentially, the understanding of value is changed: from value-in-exchange to value-in-use. First, in contrast to traditional, technology-driven development, the focus is more on the holistic customer journey. It also covers the process on how the service is taken into use rather than just the usage. Second, services need to address various customer needs and different behavior and thus, the value of a certain feature varies between different users. In traditional services, this can be addressed through front stage employees of the service provider, but in digital services, there is not human intelligence to adapt to different customer needs. Third, technical design sets clear borders of the scope. However, these do not exist in the same manner in the customers' mind. Consequently, the user experience might be

impacted negatively, since only parts of the customer journey are supported. While technical implementations will always have borders, it is important to understand the whole customer journey, in order to design the best possible solution.

Furthermore, introducing SD to software engineering requires a change in understanding of design and its role in the development process. User interface designers are often brought in late in the software engineering process. Their role is mostly the visual design and user experience of the user interface. Often, they are not involved in defining the problem that the software is solving. This was also the case of MSS – the project started off with a technical prototype, rather than starting with a user-centered perspective. However, the design of user interfaces requires a holistic understanding of the stakeholder needs, domain, and the problem [39]. Moreover, when starting with a technical prototype, as in the case of MSS, there is a risk that the existing technical prototype limits the possible solutions. For digital services, software plays a significant role and discarding a previously developed technical prototype and software architecture is often not a desired outcome. One risk of this technology-driven approach is that the wrong problem might be solved. While service design can nevertheless help to better understand users' needs and improve the overall user experience, the change is likely to be of incremental nature rather than radically new due to the constraints of the existing technical prototype.

Service design focuses on discovering and exploring the underlying problems of the customers first, before starting to think about solutions. A practical way to bring service design into the process is the usage of a variety design methods, such as explorative prototypes. In contrast to evaluative prototypes, traditionally used in software and usability engineering, their focus is on exploring the problem rather than evaluating the solution. Furthermore, design artifacts in software engineering often refer to the software specification and are mostly technical. Service design artifacts cover more holistically the customer experience over only specific requirements and can take a variety of forms. These can support better understanding of the problem and solution for different groups of stakeholders.

Service design originally focuses on traditional services, rather than digital services. Thus, SD does not have any methodology for the implementation of software and thus, it needs to be combined with models for software development, such as agile, to actually implement the service concepts. Moreover, SD visualizations and prototypes do not provide detailed specification to developers. Consequently, there is currently a gap in moving to the actual technical implementation. This issue could partially be overcome by including developers already in the development of the service concepts.

6.2 Prototyping in Digital Service Design

The technical prototype was built more closely to traditional software engineering approach [40], i.e. the requirements were documented in detail and the technical prototype was evaluated against these requirements. The purpose of the technical prototype was to evaluate the technical solution. In contrast, the service design prototypes were built without specified requirements. Furthermore, their purpose to

explore user needs rather than evaluate a solution. This has also been referred to as a shift from 'specification-drive prototypes' to 'prototype-driven specification' [41].

Using prototypes in the session with the potential users proofed to be beneficial for gaining better understanding of users needs as well as discovering which aspects need to be explored further. In addition, presenting different design alternatives to the potential users, especially low-fi paper prototypes, facilitated an open mindset and open discussion with the users, which allowed proofing some assumptions to be wrong.

While prototyping is generally considered central to service design, there is little guidance on the choice of methods to use for a specific service. Existing frameworks for prototyping of services [6], [33] can guide the prototyping process. However, they do not provide any concrete methods for implementing prototypes depending on the type of service and purpose of the prototype. Thus, the success or failure of prototypes depends largely on the designers' choices. For MSS, paper prototypes were used in order to encourage more open feedback. Furthermore, different options were presented, in order to avoid solely affirmative feedback for the presented design. However, despite the presentation of design alternative, one challenge was to encourage the participants to create and share own ideas, and be critical. The participants seemed limited with the design alternatives. A possible approach to overcome this issue would be a workshop focusing on the creation of new ideas. The paper prototypes seemed more suitable than the interactive prototype. Thus, one success factor is to be clear about the purpose of the prototypes and choose the techniques most suitable [33].

Another challenge was prototyping the whole service experience. Enacting techniques are often suggested for traditional services, but they do not seem to be suitable for many digital services, such as MSS. However, as the purpose of the prototype test session was mainly exploration of design alternatives, the service experience can be prototyped in a more holistic manner at a later stage of the process. Nevertheless, with existing service design methods, it can be challenging to prototype the experience of digital services in a holistic manner.

7 Conclusion

This paper studies how to apply service design, and specifically prototyping, in the development of digital services to gain better understanding of users' needs. The development of digital services can benefit from service design in several ways. It supports gaining better understanding of the users' needs and developing a more holistic service experience. Furthermore, different service design artifacts facilitate the communication between different stakeholders. However, challenges exist on a practical level. These challenges include applying the value-in-use concept, adapting service design methods to digital service and practical guidelines for prototyping.

Thinking about software as a service rather than a product, mainly requires a change in the understanding of value: from value-in-exchange to value in use. This implies shifting from focusing on features to understanding the whole customer

journey, even though the needs and behavior vary from customer to customer. One challenge in the design of digital services is that there is no human, as in traditional services, to adapt for different customer needs. Another challenge is that technical design sets clear borders of the scope. However, these borders do not exist in the same manner in the mind of the customer and thus, might impact the user experience. Using service design methods, such as explorative prototyping, facilitates understanding of underlying user needs and can help to avoid receiving just affirmative feedback. However, while the plethora of different service design methods offers many opportunities, challenges arise in choosing the right method for a given purpose and context.

The study was conducted in collaboration with an industrial partner and having a technical prototype is common practice in the industry. Thus, the results of this study are applicable to similar contexts.

The results of this study focus on the benefits and challenges of applying service design in the development of digital services. A few practices were given to address these challenges. However, this study revealed several areas for further research. For digital services, in which software engineering plays a crucial role, more research is needed on the integration of service design and software engineering. Furthermore, more work is needed in order to propose concrete guidelines for applying SD to digital services, and specifically to support the choice of prototyping and other service design methods most suitable for a given context. Furthermore, the special characteristics of digital services need further clarifications.

Acknowledgements. We acknowledge the financial support of TEKES as part of the Need for Speed (N4S) program of DIGILE.

References

1. Lindberg, T., Meinel, C., Wagner, R.: Design thinking: A fruitful concept for IT development? In: Meinel, C., Leifer, L., Plattner, H. (eds.) Design Thinking: Understand - Improve - Apply, pp. 3–18. Springer, Heidelberg (2011)
2. Ostrom, A.L., Bitner, M.J., Brown, S.W., Burkhard, K.A., Goul, M., Smith-Daniels, V., Demirkan, H., Rabinovich, E.: Moving forward and making a difference: Research priorities for the science of service. Journal of Service Research 13(1), 4–36 (2010)
3. Brown, T.: Design thinking. Harvard Business Review 86(6), 84–92 (2008)
4. Liedtka, J., Ogilvie, T.: Designing for growth: A design thinking toolkit for managers. Columbia University Press, New York (2011)
5. Wetter Edman, K.: Service Design – A conceptualization of an emerging practice, Licentiate thesis, University of Gothenburg, Sweden (2011)
6. Blomkvist, J.: Conceptualising prototypes in service design, Licentiate thesis, Linköping University, Sweden (2011)
7. Zeithaml, V., Parasuraman, A., Berry, L.: Problems and strategies in services marketing. The Journal of Marketing 49(2), 33–46 (1985)
8. Lovelock, C., Gummesson, E.: Whither services marketing? In search of a new paradigm and fresh perspectives. Journal of Service Research 7(1), 20–41 (2004)

9. Segelström, F.: Visualisations service design, Licentiate thesis, Linköping University, Sweden (2010)
10. Vargo, S.L., Lusch, R.F.: Service-dominant logic: continuing the evolution. Journal of the Academy of Marketing Science 36(1), 1–10 (2008)
11. Smith, J., Colgate, M.: Customer value creation: a practical framework. Journal of Marketing Theory and Practice 15(1), 7–23 (2007)
12. Vargo, S.L., Akaka, M.A.: Service-dominant logic as a foundation for service science: Clarifications. Service Science 1(1), 32–41 (2009)
13. Meroni, A., Sangiorgi, D.: Design for services. Gower Publishing Limited, Farnham (2011)
14. Lassila, A., Jokinen, J., Nylund, J.: Finnish software product business: Results of the national software industry survey 2006. Centre of Expertise for Software Product Business, Espoo (2006)
15. Cusumano, M.A.: The business of software: What every manager, programmer, and entrepreneur must know to thrive and survive in good times and bad. Free Press, New York (2004)
16. Williams, K., Chatterjee, S., Rossi, M.: Design of emerging digital services: A taxonomy. European Journal of Information Systems 17(5), 505–517 (2008)
17. Cho, E.: Interpersonal interaction for pleasurable service experience. In: Proceedings of the 2011 Conference on Designing Pleasurable Products and Interfaces. ACM (2011)
18. Holopainen, M.: Exploring service design in the context of architecture. The Service Industries Journal 30(4), 597–608 (2010)
19. Holmlid, S., Evenson, S.: Bringing service design to service sciences, management and engineering. In: Hefley, B., Murphy, W. (eds.) Service Science, Management and Engineering Education for the 21st Century, pp. 341–345. Springer, Berlin (2008)
20. Buchenau, M., Fulton Suri, J.: Experience prototyping. In: 3rd Conference on Designing Interactive Systems: Processes, Practices, Methods, and Techniques, pp. 424–433. ACM (2000)
21. Stickdorn, M.: 5 principles of service design thinking. In: Stickdorn, M., Schneider, J. (eds.) This is Service Design Thinking, pp. 34–45. BIS Publishers, Amsterdam (2011)
22. Mager, A.: Service design as an emerging field. In: Miettinen, S., Koivisto, M. (eds.) Designing Services with Innovative Methods, pp. 28–43. University of Art and Design, Helsinki (2009)
23. Miettinen, S.: Designing services with innovative methods. In: Miettinen, S., Koivisto, M. (eds.) Designing Services with Innovative Methods, pp. 10–25. University of Art and Design, Helsinki (2009)
24. Kimbell, L.: Designing for service as one way of designing services. International Journal of Design 5(2), 41–52 (2011)
25. Sangiorgi, D.: Value co-creation in design for services. In: Miettinen, S., Valtonen, A. (eds.) Service Design with Theory: Discussions on Change, Value and Methods, pp. 95–104. Lapland University Press, Rovaniemi (2012)
26. Wetter Edman, K.: Exploring overlaps and differences in service-dominant logic and design thinking. In: 1st Nordic Conference on Service Design and Service Innovation, pp. 201–212 (2009)
27. Holmlid, S.: From interaction to service. In: Miettinen, S., Koivisto, M. (eds.) Designing Services with Innovative Methods, pp. 78–97. University of Art and Design, Helsinki (2009)

28. Houde, S., Hill, C.: What do prototypes prototype. In: Helander, M., Landauer, T.K., Prabhu, P. (eds.) Handbook of Human-computer Interaction, 2nd edn., pp. 367–381. Elsevier Science B.V., Amsterdam (1997)
29. Blomkvist, J.: Conceptualisations of service prototyping: Service sketches, walkthroughs and live sERVICE prototypes. In: Miettinen, S., Valtonen, A. (eds.) Service Design with Theory: Discussions on Change, Value and Methods, pp. 177–188. Lapland University Press, Rovaniemi (2012)
30. Brocks Jr., F.P.: The mythical man-month: Essays on software engineering. Addison-Wesley, Reading (1995)
31. Lim, Y.-K., Stolterman, E., Tenenberg, J.: The anatomy of prototypes: Prototypes as filters, prototypes as manifestations of design ideas. ACM Transactions on Computer-Human Interaction 15(2) (2008)
32. Coughlan, P., Fulton Suri, J., Canales, K.: Prototypes as (design) tools for behavioral and organizational change: A design-based approach to help organizations change work behaviors. Journal of Applied Behavioral Science 43(1), 122–134 (2007)
33. Passera, S., Kärkkäinen, H., Maila, R.: When, how, why prototyping? A practical framework for service development. In: XXIII ISPIM Conference (2012)
34. Tassi, R.: Service prototype (2009), http://www.servicedesigntools.org/tools/24 (accessed: October 07, 2014)
35. Peffers, K., Tuunanen, T., Rothenberger, M.A., Chatterjee, S.: A design science research methodology for information systems research. Journal of Management Information Systems 24(3), 45–77 (2007)
36. Yin, R.K.: Case study research, 2nd edn. Sage, Thousand Oaks (1994)
37. Stickdorn, M., Schneider, J.: This is service design thinking. BIS Publishers, Amsterdam (2011)
38. Tassi, R.: Service design tools, (2009), http://www.servicedesigntools.org/ (accessed: October 07, 2014)
39. Heiskari, J., Kauppinen, M., Runonen, M., Männistö, T.: Bridging the gap between usability and requirements engineering. In: 17th IEEE International Requirements Engineering Conference, pp. 303–308. IEEE (2009)
40. Ramesh, B., Cao, L., Baskerville, R.: Agile requirements engineering practices and challenges: An empirical study. Information Systems Journal 20(5), 449–480 (2007)
41. Schrage, M.: Cultures of prototyping. In: Winograd, T. (ed.) Bringing Design to Software. ACM Press, New York (2006)