Please quote as: Comes, D. E.; Evers, C.; Geihs, K.; Hoffmann, A.; Kniewel, R.; Leimeister, J. M.; Niemczyk, S.; Roßnagel, A.; Schmidt, L.; Schulz, T.; Söllner, M. & Witsch, A. (2012): Designing Socio-technical Applications for Ubiquitous Computing - Results from a Multidisciplinary Case Study. In: Distributed Applications and Interoperable Systems (DAIS 2012), Stockholm, Sweden, 2012. Lecture Notes in Computer Science. Springer, Berlin / Heidelberg, S 194-201.

Designing Socio-technical Applications for Ubiquitous Computing*

Results from a Multidisciplinary Case Study

Diana Elena Comes [1], Christoph Evers[1], Kurt Geihs[1], Axel Hoffmann[2], Romy Kniewel[3], Jan Marco Leimeister[2], Stefan Niemczyk[1], Alexander Roßnagel[4], Ludger Schmidt[3], Thomas Schulz[4], Matthias Söllner[2], and Andreas Witsch[1]

[1] Distributed Systems Group, Kassel University, Germany
[2] Information Systems, Kassel University, Germany
[3] Human-Machine Systems Engineering, Kassel University, Germany
[4] Public Law particulary Environmental Law and Technology Law, Kassel University, Germany
{comes,evers,geihs,axel.hoffmann,r.kniewel,leimeister,niemczyk,
a.rossnagel,l.schmidt,t.schulz,soellner,witsch}@uni-kassel.de

Abstract. A major challenge for ubiquitous system design is creating applications that are legal-compatible and accepted by their intended users. Today's European data protection principles contradict the ideas of ubiquitous computing. Additionally, users have to deal with unconventional interaction concepts leading to a low amount of trust and acceptance in such systems. Current development approaches do not sufficiently cover these concerns, as they do not systematically incorporate expertise from the relevant disciplines. We present a novel development approach for ubiquitous systems that explicitly addresses these concerns. Our primary task was to manage the increased number of stakeholders and dependencies, respectively conflicts between requirements of the particular disciplines. The approach incorporates predefined artifacts and a defined workflow with responsibilities, as well as suggesting how to develop mutual understanding. We apply this multidisciplinary approach to develop the ubiquitous application Meet-U.

Keywords: Ubiquitous Computing, Social Acceptance, Law, Usability, Trust.

1 Introduction

Since Weiser introduced the idea of ubiquitous computing (UC) considerable research has been done in this field and UC applications have then been developed [1]. UC applications use sensors to gather information of heterogeneous environments, derive context to be able to adapt to changes, and dynamically integrate services to increase

* The authors cooperate in the VENUS project of the multidisciplinary Research Center for Information System Design (ITeG), http://www.uni-kassel.de/eecs/en/iteg/venus. They are listed alphabetically.

K.M. Göschka and S. Haridi (Eds.): DAIS 2012, LNCS 7272 , pp. 194–201, 2012.

user experience. Additionally, devices are constantly getting smaller and are integrated into ordinary objects, becoming a commodity. Affordable mobile data connections connect people to data networks anytime and anywhere. Surrounded by deeply integrated information technology, users are overwhelmed by the increasing complexity of such systems. Furthermore, they are incapable of anticipating all consequences of technology use, such as emerging data protection[1] concerns. While technological challenges for the implementation of UC systems have already been widely discussed, many challenges regarding the social acceptance of UC applications are still unsolved. The development approaches by Janzen et al. [2], and Resatsch et al. [3], e.g., do not account for social issues like law, usability, and trust at all. The approach described in DIN EN ISO 9241-210 does not provide specific techniques that should be applied in single development phases. The ETHICS approach [4] includes several important aspects, but lacks the incorporation of legal issues. Thus, designers need further guidance, e.g., in terms of methods and techniques. To solve this problem, we propose a multidisciplinary development approach, including methods from law to ensure legal-compatible technology design[2], from usability engineering to develop a user interface that fits the specific requirements of UC applications, as well as from trust engineering to increase user acceptance of the developed applications. The methods and theories have been combined for the development of the mobile UC application Meet-U.

2 Requirements for the Design of UC Applications

UC applications support the user during his everyday life due to the use of sensor information to perform reasoning and adaptation [1]. The resulting requirements for the development process are described in the following. Apart from the opportunities which UC offers, it also entails considerable legal risks. Particularly, the data protection risks emerging from, e.g., ubiquitous elicitation, transmission and usage of personal data or from ubiquitous monitoring have to be considered [5]. Therefore, important approaches to solutions in the scope of designing UC applications are, e.g., data protection by technology or freedom of action supporting architectures.

A challenge in UC is the creation of an unobtrusive user-friendly interface. Such an interface should not require specific skills or training. Additionally, an adaptive interface which can still be controlled by the user is required. If the system should autonomously adapt to an unexpected state, resulting in an unexpected behavior, the user might be negatively affected. Therefore, usability is a crucial issue in the development process, and expertise needs to be incorporated.

The on-going development of UC technologies will make applications more and more complex, thus enhancing the importance of trust is an important mechanism for

[1] The European law pursues the *data protection* approach which is transposed into the European data protection directive 95/46/EC.

[2] *Legal-compatible* technology design aims at the greatest possible compliance with higher-order legal goals. This differs from legal-conform technology design which refers to the prevention of lawlessness.

perceived complexity reduction to the user. Furthermore, research on technology acceptance already emphasized that trust is a key determinant of technology adoption and usage [6]. Concerning unknown technologies and applications, the initial trust of the user is crucial. Creating this initial trust during the development of UC applications is an additional challenge in UC development.

3 Development Proposal for UC Applications

This section provides an overview of the phases and activities of our initial development proposal [1] and the methods being applied. The core of the development proposal is an iterative development approach that consists of analysis, conceptual and software design, as well as implementation and evaluation.

First activities in the development process include defining appropriate goals and writing application scenarios in the demand analysis in order to establish a multidisciplinary understanding of the purpose of the UC application. Furthermore, the scenario includes a business model for the envisioned UC application.

The extended scenarios are used afterwards to elicit requirements. Our approach considers conventional requirements that are generally used in system development. It incorporates furthermore methods for acquiring expert requirements. In order to acquire functional requirements from law, the first three tiers of the KORA method [7] have been used. This procedure is analogously accomplished for usability [8] and trust [9]. After the requirement analysis, the requirements need to be joined. Therefore, the procedure of the EasyWinWin-method is adapted to our special needs. The result is a shared requirements document with negotiated requirements.

The requirements document is the basis for developing a consistent concept design. In this stage the artifacts: use cases, application workflow, screen design and backend architecture are elaborated upon. There are persons in charge for every step which consult the experts in case if the person in charge is not able to treat single requirements. Experts continuously review the resulting artifacts. If necessary, new solutions get elaborated in multidisciplinary collaboration.

The implementation of the application is planned and conducted during the stages of software design and implementation. During the iterative implementation, prototypes are built and then regularly tested against the negotiated requirements. The results are used to modify the application design. The application is evaluated using a combination of methods established within the involved disciplines.

4 Case Study: Meet-U

The development proposal of the previous section has been applied and refined during the development of the smart mobile application Meet-U [10]. Its goal is supporting people to organize meetings with friends that take place at public or private events, such as movies at cinemas or birthday parties. We designed Meet-U to be a ubiquitous application that supports its users in every situation by adapting to context changes and avoiding distracting users from their activities. Our first version of Meet-U has been developed to demonstrate different self-adaptive behaviors of mobile ubiquitous

applications and considers only requirements and restrictions from computer science. Results of a usability evaluation with users and a legal evaluation show that the first version neither satisfies the user needs, nor is it legal-conform. Besides, adaptivity is a helpful feature in ubiquitous environments, but it is not trustworthy. To turn Meet-U into a legal-compatible, user-friendly, and trustworthy application we utilized our multidisciplinary development approach.

4.1 Demand Analysis

A brief description of Meet-U‘s main goal to support meeting friends builds the basis for further development activities. Moreover, the three central user goals have been considered: planning an event, navigating to an event, and taking part at an event. From this starting point, three different scenarios enriched with personas have been elaborated upon. The scenarios show a typical set of user interactions with the planned system. They are described from the perspective of a user and the level of abstraction does not contain any technical details. They serve as a basis for identifying tasks the user might accomplish and as foundation for the requirements analysis and interface design. We comprehensively described each user type by a persona, clarifying the users' needs. Knowledge about potential users simplifies acquiring requirements regarding usability and user trust. Personas turned out to be additionally helpful for the juridical assessment, e.g., to identify whether the system has to be used in a private or professional context. The demand analysis is completed by a business model. This ensures marketability at the first stage of development. We thus identified all possible conflicts between an economic business model and the affected laws. An unlawful business model blocks a legal-compatible application design. We propose expert reviews to ensure at least legal-conformity. We used the artifacts scenarios, personas and business model for a first validation with potential users. With the feedback we could adjust the artifacts to real user needs and expectations.

4.2 Requirements Management

After the demand analysis, we elaborated upon the requirements for the further development process. Accordingly, we elicited conventional requirements as well as expert requirements in order to realize legal provisions and promote usability and trust. The emerging functional requirements were completed by the requirements we obtained in the validation during the demand analysis. We formulated the requirements in a technical, but generally understandable language. We had to pay attention to the fact that requirements are formulated without offering concrete technical solutions. However, in our multidisciplinary collaboration we were aware of the challenge that in addition to conventional requirements, expert requirements would also influence the development. These expert requirements avoid domain specific terminology to facilitate mutual understanding.

In law, the difficulty insists that legal norms rarely contain functional requirements. Even usability and trust do not primarily focus on the system’s capabilities. To gain functional requirements from law, we used KORA; for usability and trust, we used KORA based methods. This approach assures traceability to their sources. Additionally, the linguistic change from the respective terminologies to a more

general understandable language is provided. To acquire requirements from law, we identified legal provisions from fundamental rights. These provisions were concretized to legal criteria which contain relations to technical functions as well as to the legal and social aspects. In the last step, the criteria were translated to functional requirements.

Usability considers norms such as DIN ISO 9241-110 to define requirements. Different provisions were derived, which got concretized, as proposed by the KORA method. Usability criteria contain relations to technical functions as well as to the usability aspects. They describe abstract problem solutions in relation to the provisions, but do not exhibit any connection to a certain technical problem-solving approach. In the last step, the criteria were translated to functional requirements.

Trust supporting requirements were derived based on the trust conceptualization of [11]. According to this, three dimensions for user trust in a system like Meet-U were identified: performance, process and purpose. The performance dimension reflects the capability of the system in helping the user achieve his goals, the process dimension indicates the user's perception regarding the degree to which the system's algorithms are appropriate, and the purpose dimension mirrors the user's perception of the intentions that the designers have of the system. These dimensions are themselves formed by several antecedents. The antecedents can be interpreted as under-specified functional requirements, and thus are translated into functional requirements.

About half of all Meet-U requirements were expert requirements, pointing to the impact of the involved experts. The other requirements were conventional requirements. Some of the requirements were congruent, because different stakeholders came up with requirements that meant the same. Other requirements were conflictive. Thus the resulting requirements had to be negotiated in a workshop by all stakeholders. A primary step of the workshop served to enforce a mutual understanding of all requirements. Therefore, we identified terms which could have different definitions in the involved disciplines. These terms were redefined to a consistent vocabulary, and summarized in a glossary. In the following steps congruent requirements were identified and formulated as one common requirement and in case of conflictive requirements we tried to work out a joint solution. If such a solution could not be found, the requirements with lower priority were cancelled. The unsorted requirements were, if possible, grouped to function blocks. The sorted requirements and the glossary were combined to a requirements document.

4.3 Concept Design

The basis for the concept design of Meet-U was provided by the requirements document. The concept design included the graphical user interface (GUI) and the backend software architecture, and was formed in iterative steps. The GUI design was mainly conducted by the usability expert who processed the following steps: gathering all data and functional elements, developing the information architecture and functional structure of the interface, and finally designing the screen design incorporating standard wire-frames. Functional and data elements represent information and functionality promoted to the user in the interface. Data elements are the atomic units acted upon when using a system, for example: an event, a contact, the user profile, and a notification. To illustrate relationships between data elements, we employed a

concise site-map. This shows, for example, that the data element *private event* contains the data element *participants list*, which contains the *contact profile*. So the site-map facilitates the mutual understanding for the expert reviews. Furthermore, functional elements are operations that can be carried out by data elements and their representations in the interface, e.g., selecting contacts from a list in order to create an invitation list, or creating or deleting an event.

We transferred the initial information model and functional structure based on the use cases illustrating the user interactions in the interface to solve a particular task. In our flow charts, the data elements and functional elements working as connectors between screen elements were shown. The flow charts were described in high level of detail, in order to allow expert reviews. As the use cases did not refer to a certain interface element, we had to make a translation step, supported by the perspective of usability. The interaction design considered the goals of the users and common mental models of solving the tasks. Further, if there was a cluster of related user needs, one screen was considered that incorporated all these together. A smartphone specific GUI-standard was considered as archetype users would expect it. We enhanced the information architecture and functional structure by legal and trust aspects of the requirements document. For example, as required from law, we placed the legal notice in a section that is always accessible in at most two clicks. Following this, we applied platform specific design guidelines for wire-frames and visual screen design. Additionally, the screen design of Meet-U was enhanced by trust related requirements. For example, the user could access an explanation why providing a specific information in his profile was needed, and what would be the result of not providing this information. For instance, the users were informed about the relevance of specifying their interests in order to receive adequate event recommendations and they were explained why a certain recommendation was given.

The final step was defining the back-end architecture design. In this step, we elaborated sequence diagrams, entity relationship models, and identified software components. The obvious components were the mobile device, the back-end server, and external services. The sequence diagrams were derived directly from the use cases and visualize the data flow between the software components. These artifacts were reviewed by the legal experts, in order to identify weak spots like sending the current GPS position and the destination to a navigation service. Instead of sending the current position and implicitly the IP-address to the external service, we built a more legal-compatible solution which uses the Meet-U server as a proxy to impede the identification of users. A mutual understanding was required to perform a review. Thus, we annotated the communication in the sequence diagram to display which information were transmitted. Furthermore, we developed a user-centered adaptation design. Each self-adaptation of the system could be canceled by the user in respect to the expert requirements. The use cases, flow charts, screen designs, and back-end architecture form the design concept.

5 Discussion

In the following, we describe our refinement while applying the approach, which was necessary regarding the predefined artifacts, the workflow and responsibilities for development activities, and the facilitation of mutual understanding.

For the application scenario as first artifact in the demand analysis phase, we had to define an appropriate level of abstraction. We chose an abstraction level that illustrates how the user interactes with the application and describes the environment of the application. They would serve as a foundation for the requirements analysis, but not restrict the technology design. The final artifact in the requirements management phase contained the set of negotiated requirements. We grouped these on the basis of function blocks rather than grouping by involved disciplines. This grouping reduced the probability of overlooking functional requirements in the following activities. The flow charts were drawn in greater detail so that experts from law and trust were able to review them and contribute to a socially acceptable design concept.

The common understanding of the application and its environment was achieved by scenarios elaborated upon in the demand analysis, and which then guided the development for all disciplines. We used natural language to formulate the scenarios and in the next step to define requirements. Natural language was appropriate to facilitate a common understanding rather than discipline specific terminology and representation formats. Necessary key terms that turned out to be ambiguous between disciplines were clarified, defined, and documented in the glossary. We proposed to annotate the requirements with two types of information. The disciplines by which it was declared and the source from which it was derived to reasonable dealing with the requirements and ensured that correct experts could be consulted for further work.

The consultation of legal experts during the design of the business model is crucial. This needs to be considered in the workflow. In our experience, there is a trade-off between legal and business concerns. To be able to design a legal-compatible application at least legal-conformity of the business model needs to be ensured. Further, the short iterations during conceptual design turned out to be very effective. The expert reviews helped to enforce a socially acceptable design of the application by deliberating on possible solutions.

Using our development approach, we were able to design a second version of Meet-U, explicitly incorporating insights from legal and usability provisions and trust theory. Based on the assessment of the involved experts, the resulting application design is more social-acceptable than the first version. Nevertheless, it remains unclear whether the intended users of Meet-U appreciate the derived design, as proposed by the experts. To evaluate this with intended users, we have to implement the design concept. This will be done in an iterative process where prototypes will be implemented and evaluated by experts as described in the development proposal.

We observed that due to the involved experts, the overall effort of the approach was very high. We needed a lot of input and workshops with them during the development. Even if this was essential for our development process, it is also a weakness of our approach, because high expert effort means generally high expense. Furthermore, our approach might lead to an expenditure of time. Since we assume that the same expert requirements will become important over and over again, a possible improvement might be software requirement patterns to ease the phase of requirements management. By identifying these requirements and transforming them to patterns, expert effort will be reduced and future development processes will be shortened.

6 Conclusion

In this paper, we present a novel development approach for UC applications. This approach explicitly addresses social acceptance by integrating methods from legal research, usability engineering and trust engineering. The main task was to manage the increased number of stakeholders and dependencies, respectively, conflicts between requirements of involved disciplines. Therefore, we adopted predefined artifacts, a defined workflow with responsibilities, and suggest how to develop mutual understanding required for collaborative development. The Meet-U case study shows that our approach conquers special UC requirements due to the strong collaboration between the disciplines. According to the involved experts, the result is an application design which respects law, is more usable, as well as being more trustworthy than the first version of Meet-U.

References

1. Hoffmann, A., Söllner, M., Fehr, A., Hoffmann, H., Leimeister, J.M.: Towards an Approach for Developing socio-technical Ubiquitous Computing Applications. In: SUBICO 2011, Berlin (2011)
2. Janzen, S., Filler, A., Maass, W.: Designing Ubiquitous Information Systems based on Conceptual Models. In: SUBICO 2011, Berlin (2011)
3. Resatsch, F., Sandner, U., Leimeister, J.M., Krcmar, H.: Do Point of Sale RFID-Based Information Services Make a Difference? Analyzing Consumer Perceptions for Designing Smart Product Information Services in Retail Business. Electronic Markets 18, 216–231 (2008)
4. Mumford, E.: Effective systems design and requirements analysis: the ETHICS approach. Macmillan (1995)
5. Roßnagel, A.: Datenschutz in einem informatisierten Alltag. Friedich-Ebert-Stift (2007)
6. Leimeister, J.M., Ebner, W., Krcmar, H.: Design, Implementation, and Evaluation of Trust-Supporting Components in Virtual Communities for Patients. Journal of Management Information Systems 21, 101–135 (2005)
7. Hammer, V., Pordesch, U., Roßnagel, A.: KORA–Eine Methode zur Konkretisierung rechtlicher Anforderungen zu technischen Gestaltungsvorschlägen für Informations-und Kommunikationssysteme. Infotech/I+ G, 21–24 (1993)
8. Behrenbruch, K., Jandt, S., Hoberg, S., Roßnagel, A., Schmidt, L.: Normative Anforderungsanalyse für ein RFID-basiertes Assistenzsystem für Arbeitsgruppen. In: GfA-Frühjahrskongress, Kassel (2012)
9. Söllner, M., Hoffmann, A., Hoffmann, H., Leimeister, J.M.: Vertrauensunterstützung für sozio-technische ubiquitäre Systeme. Zeitschrift für Betriebswirtschaft (to appear, 2012)
10. Comes, D., Evers, C., Geihs, K., Saur, D., Witsch, A., Zapf, M.: Adaptive Applications are Smart Applications. In: International Workshop on Smart Mobile Applications, San Francisco (2011)
11. Söllner, M., Hoffmann, A., Hoffmann, H., Leimeister, J.M.: Towards a Theory of Explanation and Prediction for the Formation of Trust in IT Artifacts. In: SIGHCI 2011, Shanghai, paper 6 (2011)