Please quote as: Hoffmann, A.; Söllner, M.; Fehr, A.; Hoffmann, H. & Leimeister, J. M. (2011): Towards an Approach for Developing socio-technical Ubiquitous Computing Applications. In: Informatik 2011 - Informatik schafft Communities. Beiträge der 41. Jahrestagung der Gesellschaft für Informatik e.V. (GI), Berlin, Germany.

Towards an Approach for Developing socio-technical Ubiquitous Computing Applications

Axel Hoffmann, Matthias Söllner, Alexander Fehr,
Holger Hoffmann and Jan Marco Leimeister

Information Systems
Kassel University
Nora-Platiel-Str. 4
34127 Kassel
[axel.hoffmann|soellner|alexander.fehr|holger.hoffmann|leimeister]@wi-kassel.de

Abstract: The purpose of the paper is to make a step towards a development approach for ubiquitous computing application. Therefore, we answer the following research questions: first, what is ubiquitous computing; second, which challenges of ubiquitous system development poses the particular nature of ubiquitous computing; and third, how to overcome these challenges by combining development methods from different fields. Major challenges in ubiquitous application development are a) dissociation from known user-interfaces, b) end users' difficulties imagining ubicomp possibilities in participatory design settings, c) easy ubicomp application evaluation exceeds possibilities of current prototyping approaches, d) supporting user acceptance for ubicomp technologies is hence limited and e) the impact on society e.g. when introducing concealed sensors for ubicomp systems. This paper elaborates the specific challenges, analyzes to what extend existing development methods can be used to overcome these challenges, and introduces the VENUS approach for developing ubiquitous computing applications including methods for deriving requirements from law to ensure legally and socially compatible technology design, and trust to increase user acceptance of the developed applications.

1 Introduction

Inventing the term *ubiquitous computing* (shortened to ubicomp) and thus establishing a new research area [BD07], Mark Weiser [WGB99] opened his article in the Scientific American with the often cited statement: "The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it." [We91]. He named this vision ubiquitous computing. Weiser illustrated his idea with many examples, and in successive papers he tried to explain the term ubiquitous computing in greater depth [We93; WGB99]. Unfortunately, he has never precisely defined the term. That is probably why authors usually only refer to Weiser's vision of ubiquitous computing or try to define its core issues by interpreting him. The lack of explicit definition has presumably led to various understandings of

ubiquitous computing [BD07]. Therefore, we give an overview of ubiquitous computing definitions and summarize key features of ubicomp used. Based upon this synthesis, we propose our understanding of ubicomp.

The technological challenges for the implementation phase of ubiquitous computing applications have already been widely discussed [Ab99; BB02; EG01; LY02]. Thus, we focus on challenges for identifying system requirements and defining the design of ubiquitous computing systems. Here we concentrate on the question of how to ensure that the needs of the user are adequately considered. This includes activities involving the user in the development process and also activities in which the user cannot be involved in. In the latter case, his needs have to be grasped differently.

The remainder of the paper is structured as follows: first, we derive the most important characteristics of ubiquitous computing systems from a synthesis of the different interpretations of ubiquitous computing and propose of understanding of ubicomp. Afterwards, we illustrate the specific challenges for the development process and particular activities. Furthermore, we provide an overview of the VENUS approach that combines existing methods for facing certain challenges and highlight the gaps that still exist.

2 Ubiquitous Computing

According to Mark Weiser, ubiquitous technology "weaves [itself] into the fabric of everyday life until it is indistinguishable from it" [We91]. For him, "the challenge is to create a new kind of relationship of people to computers" [We93]. In line with Weiser's reasoning, Demers [De94] points out that ubiquitous technology is defined by not recognizing it as technology. Pursuing another approach, Rekimoto and Nagao [RN95] contrast ubiquitous computing to virtual reality. They argue this technology aims at creating a computer augmented real environment in which small and distributed computing devices assist and enhance interactions between humans and the real world. As stated by Weiser and Brown, ubiquitous technology is deeply embedded in our daily life and it is considered as a new approach to "fitting technology to our lives" [WB96]. Specifically, Abowd makes out three common features of ubiquitous computing, which are *transparent interfaces and interaction*, *context awareness* and *automated capture* [Ab99; AM00]. Lyytinen and Yoo agree with Demers' definition but also emphasize the challenges that "originate from integrating large-scale mobility with the pervasive computing functionality" [LY02]. Corresponding to Weiser's assessment, Schmidt is more specific in his definition and regards ubiquitous computing as "the phenomenon of interacting in context with artifacts and environments that are interwoven with processing and communication capabilities" [Sc02]. By contrast, Robinson, Vogt, and Wagealla [RVW05] pursue a rather abstract and somewhat different approach because they consider ubiquitous computing as a user-centric methodology that is just a means to a purpose. Bell and Dourish [BD07], however, define ubiquitous computing more generally by visions of a technological future. They stress the lack of explicit definition that had led to various understandings [BD07].

2.1 Key features of Ubiquitous Computing

Many authors consider the *focus of attention* as fundamental in their works. Basically, ubiquitous technology aims at lowering the user's technological awareness by providing natural interfaces and intuitive user guidance [Ab99]. In this regard, conventional technology is seen as a barrier [Ab99] because it must become second nature to the user to comply with the requirements of ubiquitous technology [De94]. As a consequence, technology disappears in the background and the user rather concentrates on the actual task in the real world [AM00]. To put it in a nutshell, only if a tool is learned well, it can disappear from the user's awareness [We91]. So, ubiquitous technology should be designed for user requirements, like task-orientation and ease of use [Sc02].

The feature *context awareness* is another crucial aspect of ubiquitous technology because it supports the user. A context aware ubiquitous device is able to sense information from the physical and computational environment [Ab99] to dynamically configure its services accordingly [LY02] and enable rapid personalization [Ab99]. To this end, the user's situation is automatically sensed by a range of recognition methods in order to assist without explicitly being instructed to [RN95]. However, the user's expectation about a system and the anticipation of the reaction of it highly depend on the situation, environment and prior experience [Sc02].

Only Abowd [Ab99] and Abowd and Mynatt [AM00] name the feature *automated capture*, which means the permanent capture of the environment to allow users access past situations. While the system is waiting in the background, always ready for action, the user can get support whenever necessary.

According to many authors and users, *simplicity* is one key to success. It therefore plays a prominent role in the design of ubiquitous technology [AM00]. Today's high-tech society increasingly grasps for a reduction of complexity in computing operations. So, ubiquitous technology's initiatives are to effectively make the complex mass of technology transparent to the user, especially to those with limited technical know-how [RVW05]. The presence and a high a level of ubiquitous technology in our environment will make everyday life easier and obtaining information becomes trivial [We91]. New technology, like natural interfaces [AM00], and implicit input [RN95] contribute to a general ease of use. Still, maintaining simplicity and control simultaneously remains one of the major concerns ubiquitous technology research faces.

Today, we can observe the rapid emergence of an infrastructure that enables us mobile computation in nearly every place of the world. *Mobility*, in this context, is the capability to access computing services everywhere [LY02] and yet to work with familiar user interfaces and applications [RVW05]. While users shift between different activities and environments, the available computing resources need to dynamically adapt [AM00]. Yet, this inevitably requires the smooth networking together of devices in an environment [De94] and issues the challenge to combine large-scale mobility with pervasive computing [LY02].

The research field of ubiquitous technology also requires taking *communication* and *connectivity* into account. Ubiquitous technology not only tries to connect physical and

virtual worlds [AM00] by bidirectional communication between devices and the environment [LY02], but also poses further challenges in connecting hardware and software [We91]. Idealistically, there should be seamless interoperation between devices and homogeneity in communication [BD07].

Today's hectic high-tech society lets *implicit input* gain in importance because it minimizes user intervention in everyday life [AM00]. A ubicomp system can perceive the user's interaction with the physical environment and assess the overall situation [Sc02]. Anticipating the user's goal's, the device is able to assist in further processes without explicitly being instructed to [RN95] or even perform tasks autonomously [AM00].

Table 1 summarizes which authors highlight which key features of ubiquitous computing.

Source	focus of attention	context awareness	automated capture	simplicity	mobility	communication / connectivity	implicit input
[Ab99]	●	●	●	-	-	●	-
[AM00]	●	●	●	●	●	●	●
[BD07]	-	-	-	-	●	●	-
[De94]	●	-	-	-	●	●	-
[LY02]	●	●	-	-	●	●	-
[RN95]	●	●	-	●	●	●	●
[RVW05]	-	●	-	●	●	●	-
[Sc02]	●	●	-	-	●	-	●
[We91]	●	●	-	●	-	●	-
[WB96]	●	-	-	-	-	●	-
[WGB99]	-	-	-	●	-	-	-
● mentioned by author - not mentioned by author							

Table 1. Key features of Ubiquitous Computing

2.2 Definition

As seen in the section before, many authors refer to the features *focus of attention* and *context awareness*, which are both unique to this technology. Users enormously profit from devices that are capable of obtaining information from the environment as these

devices can configure accordingly and let the user focus on the actual task. Still, only some authors mention *simplicity* and *implicit input* although these features also play a major role. As for every technology, simplicity can be a key to success because most people nowadays fail to make use of advanced technology due to its complexity. In this respect, *implicit input* simplifies certain tasks by autonomously performing them. Aside from that, it is *mobility* that enables us ubiquitous access to information and computing services everywhere, which is another crucial aspect. The feature *automated capture* is undoubtedly very functional, yet, it is not a major aspect of this technology.

In conclusion, we see ubiquitous computing as a highly embedded technology that obtains and processes information from the environment. It can adapt to various situations and configure its services autonomously in order to assist and enhance interactions between humans and the real world. To this end, ubiquitous technology even uses implicit input to reduce the level of interventions. Summarizing, ubiquitous computing is the cooperation of IT-artifacts in the environment to support the user with customized services on demand, while the interaction of the user with the IT-artifacts is partly implicit.

3 Challenges for Ubiquitous Systems Development

In this section we present five challenges in the development of ubiquitous computing systems, which are linked to the ubicomp characteristics described in the sections before. These challenges were determined by a review of ubicomp projects and academic publications as well as affirm our own experiences. Albeit these challenges can also be encountered in other domains, they are typical of ubicomp and major issues in ubicomp development projects. Most of the challenges originate from the ubicomp interaction paradigm.

- *Dissociation*: In general it is hard for the user to dissociate from known user interfaces, because they tend to stay in their known and limited mindset.
- *Imagination*: Transparent or even invisible user interfaces demand more imagination by users within participatory development methods.
- *Demonstration*: Early versions of new user interfaces can hardly be demonstrated due to the fact that they, if invisible, can only be demonstrated by showing their functionality.
- *Acceptance*: Ubicomp, as an unknown technology to the user, needs to be accepted to become successful. Due to the insignificant impact of *ease of use*, the determinants for the ubicomp acceptance are slightly different compared to the often used TAM. These need to be considered during the development of ubicomp applications.
- *Impact on social environment*: The frequent use of sensors in ubicomp systems may cause changes in behavior and therefore would significantly affect the social environment in which it is used. Unfortunately the impact of a technology only becomes apparent during a long-term usage.

In the following sections we discuss each challenge and the reason why it is particularly interesting for the ubiquitous computing development.

Challenge 1: Dissociation

The development of future-oriented scenarios of ubicomp systems itself is a challenging task. It is most effectively done in workshops including potential users and different stakeholders [LW00]. In such workshops we can exploit the creativity enhancement of groups [CHM93]. Nevertheless, there are only a few working ubicomp systems in common use [Sc10], so the ubicomp concept is mostly unknown to people. So the workshop participants tend to stay in their known and limited mindset [SBD00] and cannot dissociate from known user interfaces. This poses a huge challenge on the development of ubiquitous computing applications because the relationship between the user and technology is said to change dramatically [We93]. Apart from the new usage possibilities, the main point for this assumption is the way of interaction between user and application, aside from established user interfaces. Thus, it is difficult for the participants to imagine the possibilities and to think of new systems that are out of scope of their current mindset [Mi05].

Figure 1. Personal Computing – Mobile Computing – Ubiquitous Computing [HHL10]

Challenge 2: Imagination

Another challenge that is related with the previous challenge is imagination. In participatory development processes it is important to put the idea of the application across to the user. Again, the limited mindset of participants causes the challenge, but the situation is different. While searching ideas before, a shared understanding of the idea by all participants is needed for further development.

Challenge 3: Demonstration

To get feedback on early versions of a system in ubicomp design usually prototyping methods are used to demonstrate the planned system. If the user interface is (almost) invisible to the user it is hard to really "show" something. This challenge needs to be addressed in a participatory development process. Low-fidelity prototyping methods usually focus on displays as a main element of the user interface. Traditional paper-based prototypes and mock-ups reach their limits when they are used for ubicomp

because for demonstrating an ubicomp system it is rather necessary to demonstrate the overall functionality than the user interface only.

Three-dimensional low-fidelity prototyping needs a lot more effort and work than it is intended by the original idea of low-fidelity prototyping. Therefore, it is more difficult to use in system development. New ubicomp systems, due to their uncommon interaction concepts, cannot be based upon existing ubicomp systems that rarely exists. This complicates the quick construction of useful prototypes. The implementation of prototypes to support the user integration will result in rising expenses, as they have to be implemented from scratch. Thus, the goal should be to reduce the effort for developers. It could be achieved by providing standardized infrastructures or prebuilt base components. These would allow the developer to concentrate on the application logic of the system so the prototype would be quickly ready for evaluation in early phases of the development process.

Challenge 4: Acceptance

A well-known model to describe what drives users to accept and use new technologies is the Technology Acceptance Model (TAM) [Da89]. In its original form it names the factors *perceived usefulness* and *perceived ease of use* as the main antecedents of acceptance. Due to the calm characteristic [WB96], ease of use was found to be less important for ubicomp [Sp08]. Assuming that systems worked invisibly in the background, specific skills or learning should not be required to handle ubicomp systems [Sp08]; rather other determinants for acceptance were mentioned - ones that need to be considered for ubicomp system development.

The ongoing research on the acceptance of ubicomp technology has not yet established a well accepted set of determinants. Instead, different constructs, such as perceived *risk* [Sh10; Sp08], *privacy* [Sh10; Sp08], *control* [Sp08], *availability* [Sh10], *benefit* [Sh10], and *security* [Sh10], are mentioned. As an important determinant of acceptance which subsumes some of the above, authors point out the importance of *trust* in ubicomp [La03; SBS04; Sh10; Sp08]. The ongoing development of ubicomp technologies will make the applications more and more complex, thus enhancing the importance of trust as an important mechanism for complexity reduction [Lu99]. Trust is defined as the "willingness of a party [trustor] to be vulnerable to the actions of another party [trustee, in our case the ubiquitous computing system] based on the expectation that the other will perform a particular action important to the trustor, irrespective of the ability to monitor or control that other party" [MDS95]. A party in this context can be either a person, a group of people, or technology [NST94]. This definition implies that trust is only important if the user perceives risk otherwise he would not make himself vulnerable. Furthermore, the amount of trust needed is determined by the degree of *risk* perceived [MDS95].

Another mechanism of complexity reduction is control. Researchers state that at least some minimum of control must be perceived by the user for trust to be able to develop [Mo05]. This is because the user's perceived control of a system decreases, on the one hand, when well known possibilities of interaction, such as mouse and keyboard, are

replaced by new interaction concepts. On the other hand, it has to be assumed that ubiquitous computing systems are less structured than current systems. This characteristic leads to less perceived control and using ubicomp system thus poses a higher risk to the users than do current systems, which then makes trust more important in ubiquitous computing [SW08]. Concerning unknown technologies and applications, the initial trust of the user is crucial [MCK02] because the user has not yet had his own experiences. During the development of ubicomp applications we are often faced with the challenge of creating this initial trust.

Challenge 5: Impact on Social Environment

A major challenge of ubicomp is the significant impact on the social environment in which this technology is used. The context awareness of ubicomp applications implies the distribution of sensors in the environment perceiving the current actual behavior, as well as the position and activity of users, which irrevocably has an impact on the social structure [ZSS10], no matter how unobtrusive they seem to be [BB02]. For example, an automatic ordering system for food and drinks could signal the users' alcohol consumption to the vendor, or locating the mobile phone could reveal if the user is at home or not. These social concerns need to be considered while developing ubicomp applications [CML09].

When designing ubicomp systems the high social impact should be taken into account. The prevention of possible future technology risks is needed [Ro93]. The minimum requirements for a socially responsible technology design can be found in *law*. These serve both the constitutionally guaranteed free democratic basic order of the state and the protection of fundamental rights of individual citizens. Some laws, such as the data protection legislation, contain explicit guidelines for the design of data processing information systems. In addition, there are design requirements in other laws that regulate only indirect information technology, such as in accordance with § 312e of the German Civil Code (BGB), regarding entrepreneurs fulfilling legal duties in the electronic exchange.

The consideration of legal requirements in system development primarily aims at compliance with statutory provisions. This prevents the development of an information system that is contrary to law. If systems meet legal requirements, they are designated as lawful [Ro93]. For the consideration of the legitimacy of systems in computer science, the concept of IT compliance has been established. Today, it is one of the most important challenges in the development of technical systems [KNZ08] since the disregard of relevant laws leads to penalties and legal consequences. To this end, laws are analyzed for containing direct or indirect legal requirements, which must be considered in the design of technology. Examples are the Digital Signature Act and the Data Protection Act from which we can directly obtain legally binding technical requirements. A failure of implementation could result in legal consequences.

In particular, legal compatibility refers to the achievement of social conditions and consequences of information systems with the objectives of the law [Ro93]. Therefore, the concept of legal compatibility goes beyond the concept of lawfulness by considering

a change of the normative scale. For example, for the purpose of secrecy of telecommunications, an encrypted communications technology is more legally compatible than one that is not. However, the unencrypted one is not unlawful. The legally compatible design of technology is beyond the minimum requirements of the law. The goal of law is not only to enforce restrictions on system developments but to bring optimization towards socially acceptable systems. Further, by permanently validating laws and their purposes, it is not necessary to redesign the systems in case of legislative changes. At the time of development, loopholes in detailed rules are also irrelevant.

Based on our experience, we agree with previous research in this field [TOP02] that known approaches probably fail to consider legal issues, especially with evolving regulations and laws [OA07] in an upcoming field like ubicomp.

4 The VENUS Approach for Ubicomp Application Development

To address the challenges described in the previous sections we propose the VENUS approach for the development of ubicomp applications. The core of the VENUS approach is an iterative development approach with analysis, conceptual and software design, implementation, and evaluation. We extend the approach by methods for acquiring requirements from usability, trust, and law (see Figure 2).

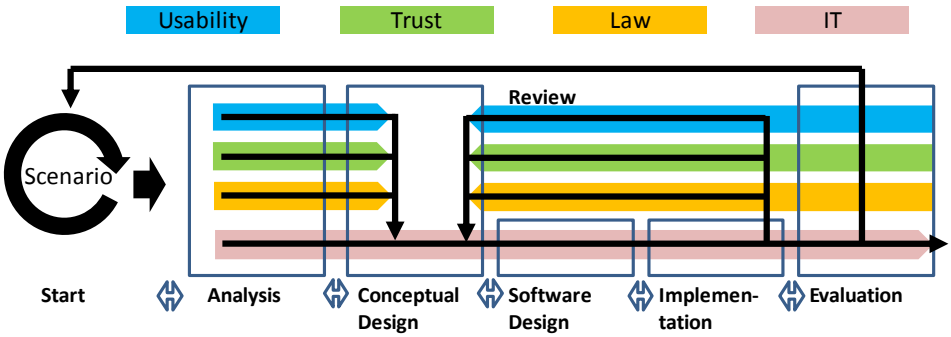

Figure 2. The VENUS development approach for ubicomp applications

The development begins with an application scenario, which ensures a common understanding. The scenario is used to elicit requirements in the analysis. These requirements are derived from IT, usability, trust, and law, which will be joined and used to build a consistent concept of the application. In software design and implementation the application is built. Prototypes should be tested against the requirements from usability, trust, and law. If necessary, the results can be used to modify the application design. Finally, the application is evaluated. With these results a new iteration of the application development can be started. Hereafter, we describe the activities and methods used in the VENUS approach.

Scenarios as Starting Point

To coordinate the analysis between usability, trust and law, it is important to develop a shared understanding of the ubicomp system. We use goals [DLF93] and scenarios [HPW02] to establish an interdisciplinary vision of the ubicomp system. Goals and scenarios are widely used in requirements engineering [Po08]. In our experience, scenarios are well suited to talk with experts from different domains [EHS11].

When creating ubicomp scenarios in workshops, it is necessary to free participants from their current mindset (see Challenge 1: Dissociation). Futuristic videos or other stimuli help relocating the participants into the "new world" (where ubiquitous computing is already realized) for the duration of the workshop [SS99]. Using such stimuli can enhance their creativity and imagination of new ubiquitous computing applications. This effect is known from science fiction movies. The viewer does not question every "unrealistic" event of the movie, because he accepts the framework of the story for the time he is watching it [SS99]. These activities help the participants to create a mental model of the future, and to imagine being a part of that future. So the participants can vision novel possibilities of ubiquitous computing. The scenario ensures a shared understanding of the application, and is the basis for acquiring requirements from usability, trust, and law in the further application development.

Acquiring Usability Requirements

A lot of research has been done in creating ubicomp interfaces and ensuring usability [e.g. DMP09; MM10; MRC01; PZ10]. We do not limit the used methods for elaborating usability. In the following section we highlight some methods that we found useful to address proposed challenges.

For usability requirements it is essential to demonstrate the scenario of the application to potential users (see Challenge 2: Imagination). To enable a quick visualization of scenarios, we use storyboards which are known from film or play. The value of a good visualization is to help participants to understand the possibilities of ubicomp systems, even if they are out of the scope of their current mindset [DLD07]. Role plays are another option for illustrating scenarios, as they enrich the concept of storyboards with interactivity [SPI04], thus making the presented ideas more real [BG00]. The scenario can be played either by the participants or professional actors with the participants watching the play. Reactions of the ubiquitous computing system are played by other actors, comparable to the wizard-of-oz-prototyping [DLO05]. It is a major challenge to free participants in the ubicomp development process from their known mindset, but this is crucial for evolving and understanding novel ideas.

Low-fidelity prototyping methods usually focus on displays as the main element of the user interface. For ubicomp it is more necessary to demonstrate the overall functionality rather than the user interface only (see Challenge 3: Demonstration). There are promising ideas of paper-based prototyping for acquiring usability requirements using three-dimensional prototypes [DLD07] that are made of paperboard or are built from other simple requisites. The resulting prototypes allow the user to get an experience of

the system analogous to traditional paper-based-prototypes. Functionalities of the ubicomp system are simulated, as it is common for wizard-of-oz-prototypes [DLO05]. The user can thus interact with the prototype, and by that can experience how the ubicomp system would operate.

The mentioned methods are only examples for acquiring usability requirements in ubicomp development, and address the challenges 2 and 3. Other suitable methods should be selected according to project structure and application type [Ni93].

Acquiring Trust-Supporting Requirements

In the VENUS approach we use a method that uses insights from trust theory to conceptualize trust supporting components for IT applications [SHA11]. The method is based upon the fact that trust as a latent variable can be influenced by its antecedents [SHH10]. Fortunately, trust researchers have identified numerous antecedents of trust [SL11] which can be influenced by specific design choices [PBM05].

The method starts with the specific goal of the application that is determined by the scenario in the VENUS approach. Trust only becomes important in situations of uncertainty [Lu99]. Thus, we need to identify situations during the interaction process between the user and the applications in which the user is confronted with different uncertainties regarding the application. Depending upon the amount of uncertainties identified it may be necessary to prioritize the uncertainties and focus on the most striking uncertainties. With help of the numerous antecedents of trust [SL11] we can derive antecedents which need to be enhanced to counter the identified uncertainties. Finally, trust supporting requirements can be formulated that enhance the identified trust antecedents.

Acquiring Legal Requirements

In the VENUS approach we use the method KORA [HPR93] that derives requirements from the purpose of law [Ro93]. This is a potentially fruitful idea for dealing with laws and regulations and is called being legally compatible. KORA follows a normative approach, in which social standards in general and legal requirements in particular, are relevant [HJH11]. In the development of technical systems - similar to the task of a judge in determining the facts of the case - developers have to derive specific legal requirements from laws and regulations. However, this task has to be carried out before there is a ready information system.

KORA starts with existing constitutional and other legal norms which can be specific legal rules. If there are no specific legal provisions applicable to the planned information system, or if they are subject to short-term changes, KORA starts with steady higher-ranked legal rules, such as can be found, for example, in the Basic Law. Based upon the purpose and the knowledge of social chances and risks inherent in the information system, fundamental legal requirements for the planned information system are developed from the constitutional and other legal norms in the first step. Hence, the

fundamental legal requirements apply to the specific project. By focusing on higher-ranked legal rules, the number of laws to be examined is narrowed down, which simplifies the selection of relevant laws. Furthermore, the differences between the laws to be considered in different jurisdictions are far greater on the lower-ranking level. If an information system is used worldwide, it must indispensably be aligned with general provisions.

Legal criteria are identified by analyzing how the fundamental legal requirements that have been developed in the first step can be qualitatively assessed with regard to the information system. The criteria rather describe abstract solutions to fulfill the fundamental legal requirements which, in principle, are legal and non-technical, but certainly can be technical. Legal criteria can also be developed on the basis of reasoning given by judges in decisions of legal cases in which the same legal norms are applied. Sometimes the criteria can already be incorporated as design demands in detailed legislature.

Technical aims for technology design are abstractions of certain characteristics which already form abstract technical requirements for the socio-technical system. As the objective of KORA is not only a lawful but also a legally compatible design of information systems, the technical aims for design are requirements which can enhance the legal compatibility. A high degree of legal compatibility ensures lawfulness for long periods of time and in different jurisdictions. If they are adopted in the system development, there will still remain considerable scope for the implementation by designers. For complex systems, further technical concretization should be carried out afterwards. The technical aims leads to legal requirements that need to be considered during the application development.

Conceptual Design and Reviews by Domain Experts

After analysis, there are requirements from usability, trust and law that need to be joined with the IT requirements. Therefore, the WinWin-method [Gr00] that supports requirements negotiation can be used. Moreover, all requirements must be drafted comprehensibly for the developers. With respect to these requirements, they can design a consistent concept of the application. The concept is used to enforce the further development process. The remaining process should be arranged according to project size and team skills. We suggest building prototypes that can be reviewed with regard to usability, trust and law experts.

In comparison to the analysis the application and the prototypes should be reviewed by experts from usability, trust, and law. This is to check if requirements are appropriately implemented, and to acquire additional requirements that need to be considered for the next prototype.

5 Conclusion

About 20 years ago, Mark Weiser invented the term ubiquitous computing. Due to the fact that he only described the vision of ubicomp without providing a definition, numerous different definitions have emerged, and many - more or less successful - ubicomp applications have been developed. While technological challenges for the implementation have already been widely discussed, many challenges regarding other aspects of ubicomp are still unsolved. In this paper, we synthesize the core elements of the different ubicomp definitions and propose a new definition for ubiquitous computing. Afterwards, we elaborate five challenges that ubicomp developers face, and provide solutions on how to overcome these challenges with the VENUS approach. Apart from usability, we integrate a method that emphasizes trust supporting requirements and a method from law that provides requirements to ensure legally and socially compatible technology design. By incorporating these components into the VENUS approach, the user acceptance of applications developed using the VENUS approach is expected to increase.

References

[Ab99] Abowd, G.D. Software engineering issues for ubiquitous computing. Proc. 21st International Conference on Software Engineering, ACM Press (1999).

[AM00] Abowd, G.D., and Mynatt, E.D.: Charting past, present, and future research in ubiquitous computing. ACM Trans. Comput.-Hum. Interact. 7, 1 (2000), 29-58.

[BB02] Banavar, G., and Bernstein, A.: Software infrastructure and design challenges for ubiquitous computing applications. Comm. of the ACM 45, 12 (2002), 92-96.

[BD07] Bell, G., and Dourish, P.: Yesterday's tomorrows: notes on ubiquitous computing's dominant vision. Personal and Ubiquitous Computing 11, 2 (2007), 133-143.

[BG00] Brandt, E., and Grunnet, C.: Evoking the future: Drama and props in user centered design. Proc. 6th Biennial Participatory Design Conference(2000), 11-20.

[CML09] Consolvo, S., McDonald, D.W., and Landay, J.A. Theory-driven design strategies for technologies that support behavior change in everyday life. Proc. CHI 2009, ACM Press (2009), 405-414.

[CHM93] Couger, J., Higgins, L., and McIntyre, S.: (Un) Structured Creativity in Information Systems Organizations. MIS Quarterly 17, 4 (1993), 375-397.

[DLF93] Dardenne, A., Lamsweerde, A.v., and Fickas, S.: Goal-directed requirements acquisition. Sci. Comput. Program. 20, 1-2 (1993), 3-50.

[DLD07] Davidoff, S., Lee, M., Dey, A., and Zimmerman, J. Rapidly Exploring Application Design Through Speed Dating. Proc. UbiComp 2007, Springer (2007), 429-446.

[Da89] Davis, F.D.: Perceived usefulness, perceived ease of use, and user acceptance of information technology. MIS Quarterly (1989), 319-340.

[De94] Demers, A.J. Research issues in ubiquitous computing. Proc. Symposium on Principles of Distributed Computing 1994, ACM Press (1994), 2-8.

[DMP09] Dillahunt, T., Mankoff, J., Paulos, E., and Fussell, S. It's not all about "Green": energy use in low-income communities. Proc. UbiComp 2009, ACM Press (2009), 255-264.

[DLO05] Dow, S., Lee, J., Oezbek, C., MacIntyre, B., Bolter, J.D., and Gandy, M. Wizard of Oz interfaces for mixed reality applications. Proc. Ext. Abstracts CHI 2005, ACM Press (2005), 1339-1342.

[EG01] Edwards, W., and Grinter, R. At home with ubiquitous computing: Seven challenges. Proc. Ubicomp 2001, Springer (2001), 256-272.
[EHS11] Evers, C., Hoffmann, A., Saur, D., Geihs, K., and Leimeister, J.M. Ableitung von Anforderungen zum Adaptionsverhalten in ubiquitären adaptiven Anwendungen. Proc. Workshops der wissenschaftlichen Konferenz Kommunikation in verteilten Systemen 2011 (WowKiVS 2011), Electronic Communications of the EASST (2011).
[Gr00] Gruenbacher, P.: Collaborative requirements negotiation with EasyWinWin. Proc. Database and Expert Systems Applications(2000), 954-958.
[HPR93] Hammer, V., Pordesch, U., and Roßnagel, A.: KORA–Eine Methode zur Konkretisierung rechtlicher Anforderungen zu technischen Gestaltungsvorschlägen für Informations-und Kommunikationssysteme. Infotech/I+ G (1993), 21–24.
[HPW02] Haumer, P., Pohl, K., and Weidenhaupt, K.: Requirements elicitation and validation with real world scenes. Transactions on Software Engineering 24, 12 (2002), 1036-1054.
[HHL10] Hoffmann, A., Hoffmann, H., and Leimeister, J.M. Nutzerintegration in die Anforderungserhebung für Ubiquitous Computing Systeme. Proc. SAKS(2010), 1-9.
[HJH11] Hoffmann, A., Jandt, S., Hoffmann, H., and Leimeister, J.M. Integration rechtlicher Anforderungen an soziotechnische Systeme in frühe Phasen der Systementwicklung. Proc. MMS 2011: Mobile und ubiquitäre Informationssysteme, Lecture Notes in Informatics (LNI) (2011), 72-76.
[KNZ08] Kiyavitskaya, N., Krausova, A., and Zannone, N.: Why Eliciting and Managing Legal Requirements Is Hard. Proc. Requirements Engineering and Law(2008), 26-30.
[La03] Langheinrich, M. When Trust Does Not Compute - The Role of Trust in Ubiquitous Computing. Proc. Workshop on Privacy at Ubicomp(2003).
[LW00] Leffingwell, D., and Widrig, D.: Managing software requirements: a unified approach, Addison-Wesley Longman Publishing Co., Inc., 2000.
[Lu99] Luhmann, N.: Trust and power, Wiley, Chichester, UK, 1979.
[LY02] Lyytinen, K., and Yoo, Y.: Issues and challenges in ubiquitous computing. Communications of the ACM 45, 12 (2002), 63-65.
[MM10] Mamykina, L., Miller, A.D., Mynatt, E.D., and Greenblatt, D. Constructing identities through storytelling in diabetes management. Proc. CHI 2010, ACM Press (2010), 1203-1212.
[MDS95] Mayer, R.C., Davis, J.H., and Schoorman, F.D.: An Integrative Model of Organizational Trust. Academy of Management Review 20, 3 (1995), 709-734.
[MCK02] McKnight, D.H., Choudhury, V., and Kacmar, C.: Developing and Validating Trust Measures for e-Commerce: An Integrative Typology. Information Systems Research 13, 3 (2002), 334-359.
[Mi05] Michahelles, F.: Innovative application development for ubiquitous and wearable computing, Hartung-Gorre, Konstanz, 2005.
[Mo05] Mollering, G.: The trust/control duality: An integrative perspective on positive expectations of others. International sociology 20, 3 (2005), 283.
[MRC01] Mynatt, E.D., Rowan, J., Craighill, S., and Jacobs, A. Digital family portraits: supporting peace of mind for extended family members. Proc. CHI 2001, ACM Press (2001), 333-340.
[NST94] Nass, C., Steuer, J., and Tauber, E.R. Computers are social actors. Proc. CHI 1994, ACM Press (1994), 72-78.
[Ni93] Nielsen, J.: Usability engineering, Acad. Press, Boston, 1993.
[OA07] Otto, P.N., and Anton, A.I.: Addressing Legal Requirements in Requirements Engineering. Proc. 15th IEEE International RE Conference(2007), 5-14.
[PZ10] Park, S.Y., and Zimmerman, J. Investigating the opportunity for a smart activity bag. Proc. CHI 2010, ACM Press (2010), 2543-2552.
[PBM05] Patrick, A., Briggs, P., and Marsh, S.: Designing systems that people will trust. Security and Usability: Designing Secure Systems That People Can Use (2005).

[Po08] Pohl, K.: Requirements Engineering, dpunkt-Verl., Heidelberg, 2008.
[RN95] Rekimoto, J., and Nagao, K. The world through the computer: computer augmented interaction with real world environments. Proc. User interface and software technology 1995, ACM Press (1995), 29-36.
[RVW05] Robinson, P., Vogt, H., and Wagealla, W.: 2005. Some Research Challenges in Pervasive Computing. Privacy, Security and Trust within the Context of Pervasive Computing, 1-16.
[Ro93] Roßnagel, A.: Rechtswissenschaftliche Technikfolgenforschung: Umrisse einer Forschungsdisziplin, Nomos Verl.-Ges., Baden-Baden, 1993.
[SBS04] Salvador, T., Barile, S., and Sherry, J. Ubiquitous computing design principles: supporting human-human and human-computer transactions. Proc. Ext. Abstracts CHI 2004, ACM Press (2004), 1497-1500.
[SBD00] Santanen, E.L., Briggs, R., and De Vreede, G.-J.: The cognitive network model of creativity: a new causal model of creativity and a new brainstorming technique. Proc. 33rd Hawaii International Conference on Systems Sciences(2000).
[SS99] Sato, S., and Salvador, T.: Playacting and focus troupes: theater techniques for creating quick, intense, immersive, and engaging focus group sessions. Interactions (1999), 35-41.
[Sc02] Schmidt, A.: 2002. Ubiquitous Computing - Computing in Context.
[Sc10] Schmidt, A.: Ubiquitous Computing: Are We There Yet? Computer 43, 2 (2010), 95-97.
[Sh10] Shin, D.: Ubiquitous Computing Acceptance Model: end user concern about security, privacy and risk. International Journal of Mobile Communications 8, 2 (2010), 169-186.
[SHA11] Söllner, M., Hoffmann, A., Altmann, M., Hoffmann, H., and Leimeister, J.M. Vertrauen als Designaspekt – Systematische Ableitung vertrauensunterstützender Komponenten am Beispiel einer mobilen Anwendung. Proc. VHB Jahrestagung(2011).
[SHH10] Söllner, M., Hoffmann, A., Hirdes, E.M., Rudakova, L., Leimeister, S., and Leimeister, J.M. Towards a Formative Measurement Model for Trust. Proc. 23rd Bled eConference(2010).
[SL11] Söllner, M., and Leimeister, J.M. Did they all get it wrong? Towards a better measurement model of trust. Proc. Academy of Management Annual Meeting(2010).
[Sp08] Spiekermann, S.: User control in ubiquitous computing: design alternatives and user acceptance, Citeseer, 2008.
[SW08] Staples, D.S., and Webster, J.: Exploring the effects of trust, task interdependence and virtualness on knowledge sharing in teams. Information Systems Journal 18, 6 (2008), 617-640.
[SPI04] Strömberg, H., Pirttilä, V., and Ikonen, V.: Interactive scenarios - building ubiquitous computing concepts in the spirit of participatory design. Personal Ubiquitous Computing 8, 3-4 (2004), 200-207.
[TOP02] Toval, A., Olmos, A., and Piattini, M.: Legal requirements reuse: a critical success factor for requirements quality and personal data protection. Proc. 10th IEEE International Requirements Engineering Conference(2002), 95-103.
[We91] Weiser, M.: The Computer for the 21st Century. Scientific American 265, 3 (1991), 94-104.
[We93] Weiser, M.: Some computer science issues in ubiquitous computing. Commun. ACM 36, 7 (1993), 75-84.
[WB96] Weiser, M., and Brown, J.: Designing calm technology. PowerGrid Journal 1, 1 (1996), 75-85.
[WGB99] Weiser, M., Gold, R., and Brown, J.: The origins of ubiquitous computing research at PARC in the late 1980s. IBM Systems Journal 38, 4 (1999), 693-696.
[ZSS10] Zeal, J., Smith, S.P., and Scheepers, R.: Conceptualizing Social Influence in the Ubiquitous Computing Era: Technologie Aoption and Use in Multiple Use Contexts. ICIS 2010 Proceedings (2010).