Observatorio (OBS*) Journal, 5 (2008), 057-077 1646-5954/ERC123483/2008 057

A Transdisciplinary Study Design on Context-aware Applications and Environments. A Critical View on User Participation within Calm Computing

Johan Criel, Laurence Claeys, Alcatel-Lucent, Belgium

Abstract

In this paper, we explore the notion of user participation in context-aware applications and environments from a transdisciplinary perspective. A critical view on user participation within calm computing is discussed and some problems are identified. The reflection of these problems in current context-aware applications and environments are described. We propose a set of recommendations and possible realizations to overcome these problems and empower users in configuring and using these applications.

1. Introduction

The last fifteen years there is intensive research in academic circles as well in the industry on the topic of context-awareness. This strong interest is also stimulated by European policy makers (Ducatel, Bogdanowicz et al., 2003), but no real breakthrough in research is perceived. If context-aware environments are so intelligent as some people maintain, and context-aware applications make the life of the people better, why are not all environments context-aware right now?

We believe that current vision of user participation within calm computing is one of the causes. To clarify our concerns we start with following simple example of a context-aware application that is often described in papers (e.g. in Meyer & Rakotonirainy, 2003): the light is switched on automatically when some conditions are satisfied, e.g. a person enters the house while it is dark. A lot of questions arise when taking a closer look to this use case. People often don't know why the light suddenly goes on, which may cause a feeling of fear and hostility. Often developers have decided when the light goes on while users cannot change it. In a lot of context-aware applications the user doesn't even has the possibilities to switch off these 'intelligent' functionalities, so that s/he just has to obey what the developers defined for her/him. Also, 'darkness' has no fixed meaning. It has a different meaning for different persons and in different interactions. At last we also doubt the usefulness of this application.

It is clear that context is a difficult topic to tackle; we assume this is because it touches upon the basic structures of interactions in the everyday life world. Together with the evolution of putting the focus on 'the user' in innovation research, both play an important role for the development of context-aware applications. We doubt if the current 'user centered design' vision puts the user at such a central stage as they often

claim. This goes together with the vision on ubiquitous computing as letting disappear (graphical user) interfaces or technology in general disappear as much as possible. In this article we discuss if this goal is the one to follow if we want context-aware applications to succeed. Maybe the vision on ubiquitous computing as initially defined by Weiser is not the way to follow completely?

In this article we take a transdisciplinary look to the evolution of context-aware applications and environments. We start with the deconstruction of the concept of 'user participation' and study how this is relating to the current trends in design of applications. In the next section we describe how these problems are reflected in current context-aware applications and environments. A brief history and some major trends of creating, configuring and using these types of applications are outlined, focusing on the shift of power relations and control. Thereafter we define some important conditions that have to be realized to make a shift in these relations and empower the user when interacting with context-aware environments. At the end we formulate some conclusions and link them to future work.

2. Questioning user participation in calm computing

When searching for the causes of the failure of a real breakthrough of context-aware applications we can determine dominant discourses and trends, which from our opinion are important influences on the process of creating context-aware applications. We focus on the empowerment of the user in the process of giving meaning to the interaction between humans and non-humans. We situate ourselves in the tradition of others who focus as well on this empowerment perspective in building context-aware applications (Gajos, Fox et al., 2002; Dey, Sohn et al., 2006).

2.1 Calm computing incompatible with feelings of fear?

The contradiction we want to discuss first is the tension between 'ubiquitous' computing which focus on the disappearance of computing and the fear of humans for technology they cannot perceive or control.

2.1.1 Dominant vision on ubiquitous computing

Nowadays humans predominantly interact with computers mediated by a (graphical) interface. What the impact of context-aware applications will be on the interface as omnipresent interlocutory space is uncertain, but it probably will loose its central stage as mediator in interactions. Mark Weiser writes about the disappearing interfaces: *"Ubiquitous computing names the third wave in computing, just now beginning. First were mainframes, each shared by lots of people. Now we are in the personal computing era, person and machine staring uneasily at each other across the desktop. Next comes ubiquitous computing, or the*

age of calm technology, when technology recedes into the background of our lives" (Weiser, 1991). It assumes that technology will be an integral part of interactions, but that the technology behind will 'disappear' and invisibly be integrated in everyday life world. The different metaphors used for context-aware applications assume this disappearing interface and invisibility of technology: ubiquitous computing, ubicomp, pervasive computing, ambient intelligence, everywhere or calm computing (Sterling, 2006). A disappearing interface is inextricable linked with an intrusive way of communication and with black boxing the technology even more than nowadays.

However, it is not because physically the computer will disappear that the logics of the computer will do so and context-aware environments will become 'intelligent'. Already in 1983 Philip J. Hayes en D. Raj Reddy, both computer linguistics, quoted two important differences between interactive computer systems and human communication, which make them fundamentally different: a) interactive computer systems do not possess the capacity to answer unanticipated circumstances; and b) interactive computer systems do not possess solutions to detect communication problems (Hayes & Reddy, 1983: 12). And we would complete this with c) interactive computer systems never work perfect. The impregnation of the life world by ubiquitous computing is limited because machines lack a certain amount of common sense (Dreyfus, 2001). Or, as Lucy Suchman writes on the difference of (human) situated actions and (machine) planned actions: *"The circumstances of our actions are never fully anticipated and are continuously changing around us. As a consequence our actions, while systematic, are never planned in the strong sense that cognitive science would have it. Plans are a weak resource for what is primarily an ad hoc activity"* (Suchman, 2005: 20). Interacting with machines is very different from interacting with humans because machines can never understand the intentions and the meanings humans give to communication acts. We cannot speak of a *general thesis of reciprocity of perspectives* (Schütz & Luckmann, 1974) in communication acts between humans and machines.

2.1.2 Fear for technology

When looking at recent research on ICT use (Broos & Roe, 2006) and the meaning of information- and communication technologies (ICT) in everyday life (Bakardjieva, 2005; Claeys, 2007), we notice that feelings of fear for ICT, but also the fear of evolutions in the network society, are related to feelings of self efficacy, and locus of control. The digital literacy level plays an important role in the feelings of control (and of fear). Without discussing the different models of digital literacy (Barton & Hamilton, 2000) we want to stipulate that without digital literacy, and without feeling of control, people will stay scared of technological developments and changes regarding their individual lives and the society they live in.

Combining the dominant vision on ubiquitous computing with the findings on causes of the feelings of fear for technology, we can already define one problem space: disappearing technology and not having feelings of control exclude each other.

2.2 User participation as empty signifier?

The development of new (context-aware) applications envisions a central role to the user. In industrial research, different participative methods are used to learn about the future users, or even to co-create applications together with users (Van Rompaey, Van Der Meerssche et al. 2005). Also in promotional material of Ambient Intelligent applications for the home, the words 'connectedness', 'control', 'easiness' and 'personalization' are predominant. But in the same work the researchers determined that, in contradiction to the discourse of 'putting the user central', almost half of the pictures used in the promotional material contained no humans but devices (Ben Allouch, Van Dijk et al., 2005). What does it mean to have the users participating in the development process? What is their impact? Has this concept become an empty signifier (Laclau & Mouffe, 1985)?

2.2.1 The origin of user participation

Since the 80's there is growing attendance for the role of the 'user' in innovation. This is translated by co-employing usability designers and ethnographers in teams that existed before only of engineers. It is not traceable who focused for the first time on the 'user'[1], but the research area known as Participatory Design (PD), from origin Scandinavian, played certainly an important role. Participatory Design is a research area that initially started from Trade Union Participation (Beck, 2001). A central concern was that workers needed to be able to participate in the means of production. Therefore representatives needed to understand new technologies to be prepared for negotiations with management (Nygaard & Bergo, 1973). Thereafter different political, and non-political, researchers focused on the development of specific techniques for involving users in design (for an overview, see: Bjerknes, 1987). In Participatory Design 'user participation' was in no way related to business issues or forecasting the market, but was only seen as a vehicle for empowerment of the user in different ways.

From social science perspective the rise of the Social Shaping of Technology (SST) research area gave a growing attendance on the role of the 'user'. Although SST is an umbrella concept for different socio-constructivist visions on technology, which rose as a reaction on the dominant technological deterministic visions, all the visions focus on the mutual shaping process of use and development, of society and technology. SST examines the politicization of technological culture, the interpenetration of materiality and

[1] As all changes in practices or theory different authors create simultaneously a new research direction.

identity and the role of users as agents of technical change (Boczkowski, 2004). More than in PD which especially deals with the changes technology brings when inserted in existing work relations, SST focuses on power relations that are inscribed or embedded in technology. This means that scripts can sustain, enforce or transform existing power relations.

2.2.2 The notion of participation

Participation has been used to refer to a wide variety of different situations by different people, and is a very contested concept. Therefore some refer to it as an *empty signifier* (Carpentier, 2007). The history and origin (and radicalism) of the concept as related to power issues is fading away under the diversity of meanings it is covered now. Servaes refers to the importance of power as central entity that must be linked to participation. As he writes: *"this 'real' form of participation has to be seen as participation [that] directly addresses power and its distribution in society. It touches the very core of power relationships"* (Servaes, 1999: 198 in: Carpentier, 2007: 87).

Carole Pateman makes a differentiation between 'partial' and 'full' participation. She defines partial participation as: *"A process in which two or more parties influence each other in the making of decisions but the final power to decide rests with one party only"* (Pateman, 1970: 70). Full participation is*: "A process where each individual member of a decision-making body has equal power to determine the outcome of decisions"* (Pateman, 1970: 1971).

2.2.3 The impoverishment of 'user participation'

Nowadays different R&I departments of IT-company (Xerox, Nokia, Philips, Alcatel-Lucent) work with transdisciplinary teams. Performing user research and organizing co-design sessions are commonly used practices in these environments; different methods are gathered under the name 'user centered design', and all of them lean on the participation of the user in the innovation process. Although the concepts are still currently fluid and under discussion, the 'user centered design' process puts users at the center of the innovation process. Within co-design the user is integrated in a very early stage of the conceptual and interface design process and the focus is on the mutual learning process of developer and user. Within user-informed design information about the user(s) is gathered before developing a design and the user is integrated at a certain moment in the design process (Geerts, Jans et al., 2007). Within user-informed design we can further distinguish e.g. interaction design (Arnall, 2006) and experience design (Forlizzi & Battarbee, 2004; Jones, 2006).

This widespread adoption of the concept 'user participation', and diversification of design methods that put the user central, does not mean that the original ideas on participation and user participation, as historical

rooted within PD, are also widely disseminated. The question even is if these design methods can be defined as 'participatory' if we look at the interpretation and meanings attached onto the concept of 'participation' as described above.

Where in PD empowerment and participation were central and political aims, the different user-centered design models are mostly not political and not concerned with power relations and empowerment. It seems like the success of 'user participation' is accompanied with an impoverishment of the concept of 'user participation'. As Beck formulates it: *"participatory design has come to include practices that share only the historical link to participation as a vehicle for empowerment. In the non-political extreme, user participation, once politically radical, has been picked up as a slogan for marketing and other uses"* (Beck, 2001: 6).

When we talk about user participation in the development of applications and refer to the definitions of Pateman, we can in 'best' cases speak of a certain form of partial participation, but in no way of full participation.

2.3 Design trends incompatible with full participation?

If the existing design trends are compatible with the vision on participation as formulated above is the next question discussed. By attaching our vision on the interpretation of participatory design onto the existing trends regarding (GUI) design.

2.3.1 The goal of getting it simple and easy

User-centered design is currently the dominant trend in design processes. Designers try to know as much as possible about their future users. However the problem is that when applications are getting tailored for the particular likes, dislikes, skills and needs of a particular target population, they will less likely be appropriate for others (Norman, 2005). Another drawback is that working with specific target groups the design tends to become quickly a strengthening of existing stereotypes. To put it simple: it makes everything pink and round for women, and blue and angular for men. This may conduct shortage to the individual preferences and reality of a liquid identity, preferences and life of users. Don Norman, who is perceived as a leading voice regarding user-centered design, questions the current direction of design. He states: "*Human-centered design is considered harmful*" (Norman, 2005).

Cecile Crutzen criticizes also the current vision on 'user friendly design'. She starts from the differentiation between 'readerly (lisible) text' and 'writerly (scriptable) text' as made by Roland Barthes (1977). In 'readerly' texts the author is an autonomous, authoritarian producer and sender, and the reader a prototyped passive consumer and receiver. The 'writerly' text, in contradiction, invites the active participation of the reader. Crutzen states that ironically, in design 'readerly' text is described as

userfriendly while 'writerly' text is referred to as unreadable and userunfriendly because they require more effort (Crutzen, 2005).

2.3.2 User participation within 'simple and easy' design

From our perspective, the vision on 'simple and easy' is not compatible with user participation, when going back to the initial meaning of participation as linked to power relations and empowerment. We assume that in these kinds of design the outcomes of decisions in the user-centered design process stay in the hands of the developers. When looking at the development process we see that the possible impact of the user on the development of applications is narrowing further the development of the application is completed. So we can in no way speak of more or less equal power relations. The involvement of the users is very limited. Mostly the existing patterns of behavior are observed during the contextual investigation (pre-application time), and the interaction with the application is studied when the application is already prototyped.
In figure 1 we show the possible impact of the user on the development of applications.

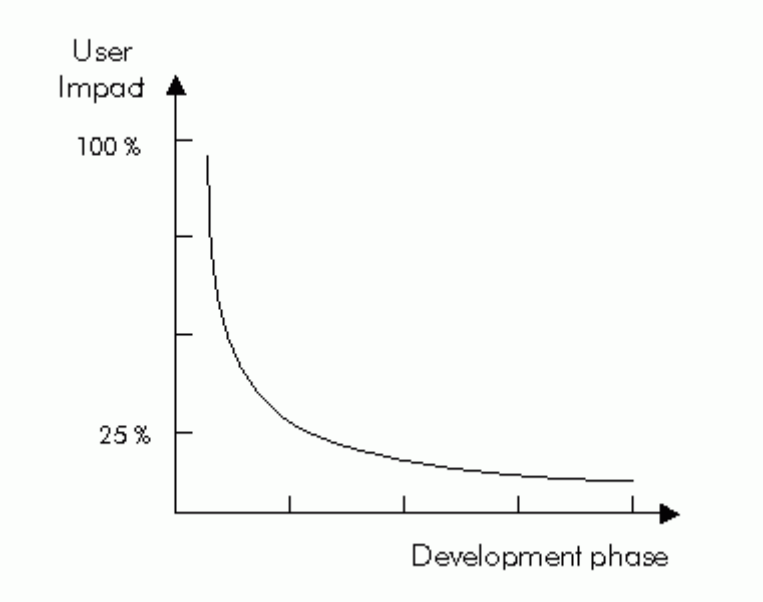

Fig. 1: Possible impact of user on application related to the development phases of applications with an 'easy and simple' design concept

When looking at the highest point, the beginning of the curve, we see it never attains the 100% level. Because the development of applications is driven by, or building upon, existing technologies. We assume that in the initial phase of the creation of a new application the user impact is always restricted by the existing technological boundaries. Even if the idea for a new application is born out of the mind of users, the application that will arise will have the limitations that are set out by the technology. When looking at the end of the curve, we see that with the closure (Bijker, Hughes et al. 1987) of the application, the possible impact of the user is reduced to almost zero. Although this is not the fact for all applications (see next paragraph), the 'easy and simple' interface design often does not accept any user impact after closure. This probably is linked to the costs for bringing changes into programs over time. The later changes have to be made within programs, the higher the costs. If a product is designed for 'simple and easy' use and only one path is possible, then it must be the 'right' path (whatsoever 'right' may mean in this case) because changing it in a later stage comes with massive costs.

3. Context and their meaning for context-aware applications

In this section, we describe how the problems stated above are reflected in current context-aware applications and environments. First, we take a short look at context definition from a transdisciplinary perspective. Afterwards we take a look at the evolution of the shift in control.

3.1 Context from a transdisciplinary perspective

Although defining concepts is a fundamental step in doing scientific research, the context definition is still under discussion. In his PhD thesis, Dey (2000) gives an extensive overview of existing definitions of 'context' (Pascoe, 1998; Schilit, Adams, et al. 1994; Chen & Kotz, 2000; Hull, Neaves, et al., 1997; Brown, 1996), which are used in the literature on context-awareness.

Dey himself quotes that context-aware applications look at the who's, where's, when's and what's of entities and use this information to determine why a situation is occurring. From this perspective he defines context as: "*Any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and application themselves"* (Dey, 2000).

Our question regarding this strong, and often used, definition is that context is more about meanings that are constructed in interaction with persons, places or objects than about entities as persons, places or objects as such. This remark can be situated within the current evolution in human sciences where political, economical, cultural and social context are taken into account for explaining social interactions[2]. This evolution must be seen in the growing critics on the (post) positivistic and empirical-analytical tradition, which started in the '60s. This evolved into more attention for contextualizing behavior when looking for explaining social behavior (Kuhn, 1962; Schütz, 1962). The central critics on this statement were that quantitative research, beside disempowering, also decontextualized the research subjects. This all in view of reaching 'objectivity' in research. But, as constructivist researchers emphasize, interactions are always contextual situated, and meaning to it is given within this (changing) context. This is related to the view on reality not as a world that consists of facts that represent objects, but as one of intersubjective constructed meanings that are defined in interaction. Context then defines interaction, but interaction is also changing under influence of context. Context is then intertwined with the view on reality and the ontological structures of the life-world, what makes it a very philosophical discussion.

[2] Typical for an interaction is that there is always an exchange of meanings. I concur with Cecile Crutzen' definition of interaction when she writes: *"We speak of interaction when there is a process of exchange of representation(s) between actors and when these actors give meaning to these representation(s) while observing and representing, in short when these actors construe meaning by means of representation(s)"* (Crutzen 2000: 44).

From an engineering point of view, and within the constraints of existing technologies, taking meaning into account is a strange switch to make. It undermines the belief in the existence of a 'model of the user's world' or a 'model of behavior'.

3.2 Control shift in development and configuration of context-aware applications and environments

This section describes how context-aware application development changed over the years and how developers, and especially users, can configure these applications. We will discuss the basics of context-aware applications, how context knowledge can be represented within computer systems and how the creation of context-aware applications evaluated over the years.

3.2.1 Basic architecture of context-aware applications

Figure 2 represents a rough architecture of a context-aware application. Two types of input can be noticed. On the one hand context data (such as location, person presence, weather…) are acquired from (hidden) sensors or software equivalents. On the other hand explicit user input is gathered through e.g. keyboards, graphical user interfaces or GUIs. Based on context data and explicit user input the application logic defines which (new) data can be inferred or which action(s) should be performed, e.g. a light which is switched or a message which is send or the notice that, when two persons are at the same location we could infer that they are near by each other. The inferred data can then be seen as new context data. We should also be aware that all user input and output has to be situated in a broader context than can be applied by computer systems, and which every user will perceive differently.

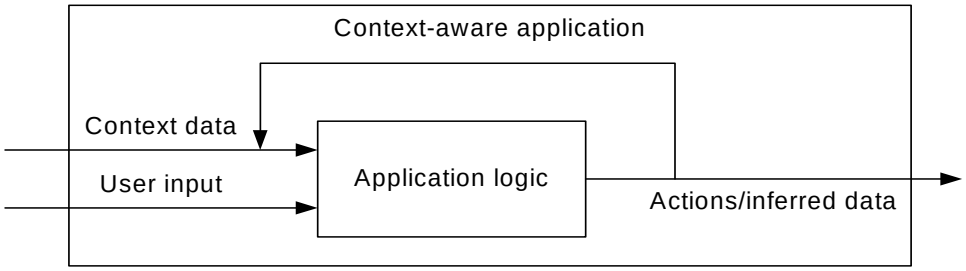

Fig. 2: Basic architecture of a context-aware application

We will focus on following technological differences between different context-aware applications: the way in which and how context data is modeled (and acquired) and how easy it is to understand and change the logic of context-aware applications.

3.2.2 Context models

To be able to deal with context in a computerized way a context model is needed. This is the way context knowledge of an environment is represented within computer systems based on the definition of context. However, doing this remains a difficult task.

Context models have involved over the years from very static, unexpressive representations to more flexible, semantic and extensible ones. In the beginning attribute-value pairs were quite frequently used e.g. in the Context Toolkit (Dey, 2000). More recent models have the capabilities to define concepts, relations and their constraints. The most common models at the moment are ontology based (Gu, Pung et al., 2005; Chen, Fenin et al., 2004; Korpip, Malm et al., 2005) although other models like object oriented (Hofer, Schwinger et al., 2003) and 4-ary predicates (Rom, Hess et al., 2002) exist.

Ontologies are very suitable for context models because they are so expressive and also because of the possibility for applying ontology based reasoning techniques (see next paragraph). Numerous representations to define ontologies exist but probably Resource Description Framework (RDF) (W3C 2000), Web Ontology Language (OWL) (W3C2004) and W3C' semantic web activities (W3C 2001) gave a boost to ontologies.

Although much technological progress has been made to create more semantic models, only little research has been performed on how models can be designed by communities (e.g. DOGMA-MESS (De Moor, 2006) and even fewer by 'ordinary users'. Almost all models are technological driven and expect that, in one or another way, it is possible to model a domain, or even worse, the world. A domain or world model is mostly based on user groups but context-aware applications are very dedicated for every user. Because of that they are more difficult or even not possible to define them. Moreover, mostly little attention is paid on what model users really need. In addition, models change over time because with every action context changes as in real life.

As a result most current models can only be used in protected research environments but not in real world situations.

Although ontologies allow a more semantic description of an environment and many authors claim to be able to model emotions and feelings of persons but forget that the meaning of context differs for every person and is modulated through interactions. A rudimentary example for instance is the difficulty to model 'having cold'. This feeling is subjective and we will probable never be able to model such entities.

3.2.3 Evolution in context-aware application creation

In the previous paragraph we discussed different ways how knowledge in a context-aware environment can be represented. This paragraph describes the evolution in building applications that can import data into a context model. This data is acquired from hardware, or virtual sensors. We also describe how actions/events can be triggered based on the data in the context model. Instead of enumerating existing context-aware applications we will focus in this paragraph on the shift of who is in control over time.

When investigating context-aware application we notice a shift from who is able to create and configure context-aware applications or, with other words, who is in control.

Before 2000 almost all sensor specific programming code was hard wired (programmed) into applications. Those applications have a lack of sensor abstraction and separation of concerns. Building these types of applications was very cumbersome and money consuming and was very difficult to reuse. The consequence was that only very few specialized developers where able to create context-aware applications.

Around the year 2000 context-aware frameworks arose which helped developers developing context-aware applications (Dey, 2000; Chen et al., 2004). Developers could use generic software components of the context frameworks which allowed acquiring context information without having to know all boring details of the hardware sensors (sensor abstraction). These components also allowed making already existing applications context-aware without having to recode the whole application (separation of concerns). Although this lowered the barrier to develop context-aware application for developers, this still doesn't give the control to the user. Moreover in most of the frameworks and applications the control logic was still hard coded by developers for a specific application. It was difficult to change for developers after going into production and almost impossible for users without having to become a general purpose developer.

The last years step-by-step research starts on how context frameworks/applications can be adapted in the way that developers can implement the logic in a more declarative way. Reasoning engines that use rules for declaring the logic play an important role. A rule consists of conditions and consequences (that can be action/events or inferred data). When the conditions are satisfied the consequences are processed. Some examples: when a person enters the room and it is dark in house (two conditions coupled by Boolean logic) the lights are automatically switched on and the music starts playing (action as consequence); or when two persons Mike and Pat are in a distance of 1 m of each other (condition) new inferred context data 'Mike is near by Path' is added to the context model.

These rule based systems or reasoners make it easier for developers but logic to define rules is still to complex too for an average user.

Last years research started on how to give users control of context-aware applications (Dey, Hamid, et al., 2004; Dey et al., 2006) and on the relevance of context topics for the users, or even better to make that the users can define the context topics that are relevant for them.

4. Defining boundaries for the development of critical user participatory context-aware applications and environments

After stating our critics on user participation and the evolutions in the development of context-aware applications and environments together, we want to formulate a non-exhaustive set of recommendations for the development of what we name 'critical [3] user participatory context-aware applications and environments'.

We identify six key conditions that from our point of view need to be fulfilled to be able to speak of 'critical' user participation in the development and configuration of context-aware applications in our urge to start to shift existing power relations in the advantage of the user. The conditions are formulated from a technological and human perspective, because developers and users both need to be accountable and knowledgeable actors.

The formulation of the conditions start from our critic that persons perceive context-aware applications at the moment as black boxes. Developers define what the output/inferred data will be for a certain input. The opening of the black box onto a certain level is inevitable for empowerment of the users. They should be able to look into the black box and define themselves for which context data (and possibly user input) a certain action should be performed. But these requirements have implications for development and use.

To demonstrate the technologically realization of our recommendations we use in our current prototypes the combination of rule engines and domain specific languages (DSL) because is probably the most appropriate way at the moment. Herby it is important that all rules are part of the context model itself. Each rule is defined and takes place within a certain context and cannot be separated from each other. This critic was also launched by Singh and Conway (2006) but then from developer's perspective.

First of all people should know about the computerized context that surrounds them. Therefore they should be aware which context can be 'sensed' and understand what it means. This doesn't mean that users should know all sensor details but at least what's possible to detect and what not. A way to tackle this problem from our perspective is that users could retrieve which context is measured in the environment surrounding them at any time and any place. Therefore (maybe separate) context-aware applications should be available to sense the environment and present the context topics that are measured in a human

[3] By using the concept 'critical' we estimate the relation to the tradition of critical theory as defined by Max Horkheimer of the Frankfurt School.

understandable way. Without access to this information users will always experience an ambient environment as suspicious or even hostile and will fear the unknown.

Secondly, users should be able to understand the logic that is applied in context-aware applications. This means that they should always be able to know why something happens. When things happen without the understanding of a person but only by the developer, the developer is determining the behavior of that person in a non-democratic way. Users for example do not appreciate when screens pop up from 'nowhere' (just look to some web pages to see that developers do it even deliberate for unhonorable purposes). A lot of applications have these problems but in context-aware applications the life of the person is affected without the feeling of direct computer interaction (and have no keyboard to beat when they are angry). It can be nice that lights are switched on without pushing a button, but people should always understand why this happens. Moreover, the reasons why something happens should not be explained in a mystic computerized way but in a human understandable one.

To give users a better understanding in the context-aware application logic, we propose that for every action that takes place in the context-aware environment a diagnosis is presented to the user that explains why something happens (if the user wishes this) in human understandable language or in a graphical way. Diagnosis is already often applied in case of errors but seldom when things go as expected. Error reporting still often happens cryptic and so not understandable by users, but diagnosis should be human understandable. Technologically backwards rule chaining could provide the means to derive the conditions when a certain action took place. Domain Specific Languages (DSL's) can then be used to translate the conditions in a (quite) human understandable way.

A third, probably one of the most simple but important, condition is that users always must be in the possibility to switch off the context-aware interactions. The ultimate control may not lay in hands of developers or others, but in hands of the persons the application is related to. Although this sounds quite logic this is most often not the case in current context-aware applications or environments. Without being a developer you cannot manage the application yourself.

A fourth condition is that users should be able to define by themselves what should happen when certain context conditions are satisfied. It could not be that developers define when and what happens without the user can intervene or adapt it afterwards. Because the inherent feature that context-aware application are very personal, developers can never create the logic needed for or wanted by the individually users. Technologically rule engines combined with DSL's (or rule editing GUI's) could allow users composing their own context-aware logic by defining their own rules without having to become a general-purpose developer. We are, however, aware that it will still be the developers who define the most basic context topics and actions that can be used by users to create their own rules.

A fifth condition, probably the most difficult one to realize, is that users should be able to define there own meaning to context topics. Meaning is so personal and evaluating in time that developers (also when applications are developed in transdisciplinary teams) never can define it for the user, although this is what often happens at the moment. Just as in the previous condition, rules with inferred data as consequence could be used to implement a very basic form of meaning. Users could for example define what they understand as 'cold' under certain conditions. This can be 5 degrees Celsius, or 20 degrees Celsius. We know that this basic example is only a start of solution to put meaning in the hands of users. We still have to take into account the constraints of technology and technology cannot handle meaning and meaning is constructed in the interactions themselves.

A last, but very important condition, is the fact that not only developers may not design from a *'view from nowhere'* or *'detached intimacy'* but from a *'located accountability'* (Suchman, 2002), but users also have to take their responsibility. Living in a networked society makes it necessary to develop some critical digital literacy, and also some critical literacy of the digital. A necessary condition to shift power relations regarding technology, and more specific related to context-aware applications, in favor of the user is inextricably linked with the will of users to take their responsibility in autonomous behaving and controlling their everyday life world where context-aware applications will possibly get integrated.

Although these conditions will never take away the feeling of intrusiveness completely and users will never full participate in the development and configuration of the context-aware logic, we believe that consequent satisfaction of these conditions will make the users more confident in context-aware environment and give them a greater feeling of being in control.

5. Case study design

Currently we are further elaborating, testing and researching the prototypes that can realize the conditions stated above. Moreover research is done in how rules can be defined by users in a tangible way. Results will be published at a later time.

6. Conclusions and future work

In this article we studied the reasons why context-aware applications and environments didn't really break through until now. We started with scrutinizing the concept of user participation and their relation to current design trends and the implication for the development of context-aware applications and environments. We formulated some recommendations with the goal to shift power relations in the

development, configuration and use of context-aware applications. We assume that without allowing users to open and configure the context-aware logic, users will never have trust into the systems, never use it and stay in a very weak power position in relation to the developer.

We are aware that this is only the beginning of research and the research on the implementation of mentioned recommendations should be evaluated. We are however aware that the fulfillment of these conditions will not take away the feeling of intrusiveness completely and users will never 'fully participate'. But when these conditions are not met we believe that context-aware environments will not break through and the real research for useful user programmed context-aware applications will never start.

[Future work will be proposed at the conference]

Acknowledgements

The work reported in this paper was performed in the context of the ITEA SmartTouch project: "Browsing Smart Objects Around You" and the ITEA AmIE: "Ambient Intelligence for the Elderly", funded by the Flemish IWT [IWT]. We want to thank the IWT for their support.

Bibliography

Arnall, Timo (2006). *A graphic language for touch-based interactions.* Paper presented at the Mobile Interaction with the Real World (MIRW 2006), Espoo, Finland.

Bakardjieva, Maria (2005). *Internet Society. The internet in everyday life*. London, Thousand Oaks, New Delhi: Sage Publications.

Barthes, Roland (1977). The Death of the Author. In D. Graddol & O. Boyd-Barrett (Eds.), *Media Texs: Authors and Readers* (pp. 166-170). Clevedon: Multilingual matters in association with the Open University.

Barton, David, & Hamilton, Mike (2000). Literacy practices. In D. Barton & M. Hamilton & R. Ivanic (Eds.), *Situated literacies: reading and writing in context*. London: Routlegde.

Beck, Eevi E. (2001). *On participatory design in Scandinavian computing research*. Oslo: University of Oslo, Department of Informatics.

Ben Allouch, S., Van Dijk, J. A. G. M. & Peters, O. (2005). Our future home recommended: A content analysis of ambient intelligence promotion material, *Etmaal van de Communicatiewetenschap*. Amsterdam, The Netherlands.

Bijker, Wiebe, Hughes, Thomas P., & Pinch, Trevor (1987). *The social construction of technological systems: New directions in the sociology and history of technology*. Cambridge, MA: MIT Press.

Bjerknes, G., P. Ehn, et al. (1987). Computers and Democracy - A Scandinavian Challenge. Aldershot.

Boczkowski, Pablo J. (2004). The Mutual Shaping of Technology and Society in Videotex Newspapers: Beyond the Diffusion and Social Shaping Perspectives. *The Information Society, 20*, 255-267.

Broos, Agnetha, & Roe, Keith (2006). The Digital Divide in the Playstation Generation. Self-efficacy, locus of control and ICT adoption among adolescents. *Poetics*.

Brown, P.J. (1996). *The Stick-e Document: a Framework for Creating Context-Aware Applications. Proceedings of EP'96, Palo Alto*, pages 259-272.

Carpentier, Nico (2007). Introduction: Participation and Media. In B. Cammaerts & N. Carpentier (Eds.), *Reclaiming the Media: Communication Rights and Democratic Media Roles*. Bristol, Chicago: Intellect.

Chen, G., & Kotz, D. (2000). *A Survey of Context-aware Mobile Computing Research. Technical Report TR2000-381*: Department of Computer Science, Dartmond College.

Chen, H., Fenin, T., & Joshi, A. (2004, March 14-17). *Semantic Web in Context Broker Architecture. .* Paper presented at the Second IEEE International Conference on Pervasive Computing and Communications (Percom '04), Washington DC.

Claeys, Laurence (2007). *Informatie-en Communicatietechnologieen in de dagelijkse leefwereld. Een interpretatieve benadering van gelijke kansen in de netwerkmaatschappij.* Universiteit van Gent, Gent.

Crutzen, Cecile (2005). Questioning Gender in E-learning and its Relation to Computer Science. Spaces for Design, Working and Learning. In R. Braidotti & A. Baren (Eds.), *The Making of European Women's Studies, Volume VI* (pp. 40-59). Utrecht: Athena.

De Moor, A., De Leenheer, P., Meersman, M. (2006). *DOGMA-MESS: A Meaning Evolution Support System for Interorganizational Ontology Engineering.* Paper presented at the 14th International Conference on Conceptual Structures, Aalborg, Denmark.

Dey, Anind K. (2000). *Providing architectural support for building context-aware applications.* Phd. College of Computing, Georgia Institute of Technology.

Dey, Anind K., Hamid, Raffay, Beckmann, Chris, Li, Ian, & Hsu, Daniel (2004). a CAPpella: programming by demonstration of context-aware applications, *Proceedings of the SIGCHI conference on Human factors in computing systems*. Vienna, Austria: ACM Press.

Dey, Anind K., Sohn, Timothy, Streng, Sara, & Kodama, Justin (2006). *iCAP: Interactive Prototyping of Context-Aware Applications.* Paper presented at PERVASIVE. Springer-Verlag (pp. 254-271).

Dreyfus, Hubert L. (2001). *On the Internet*. London and New York: Routledge.

Ducatel, K., Bogdanowicz, F., Scapolo, J., Leijten, J., & Burgelman, J-C. (2003). *Ambient Intelligence: From vision to reality*: IST Advisory Group Draft Rep.European Commission.

Forlizzi, Jodi, & Battarbee, Katja (2004). *Understanding Experience in Interactive Systems.* Paper presented at the DIS2004, Cambridge, Massachusetts, USA.

Gajos, K., Fox, H., & Shrobe, H. (2002). End user empowerment in human centered pervasive computing, *Pervasive 2002* (pp. 134-140).

Geerts, David, Jans, Greet & Vanattenhoven, Jeroen (2007). *Terminology*. Presentation at CitizenMedia meeting, 15&16 february, Leuven, Belgium.

Gu, Tao, Pung, Hung Keng, & Zhang, Da Qing (2005). A service-oriented middleware for building context-aware services. *J. Netw. Comput. Appl., 28*(1), 1-18.

Hayes, Philip. J., & Reddy, Raj D. (1983). Steps toward graceful interaction in spoken and written man-machine communication. *International Journal of Man-Machine Studies, I*(19), 231-284.

Hofer, Thomas, Schwinger, Wieland, Pichler, Mario, Leonhartsberger, Gerhard, Altmann, Josef, & Retschitzegger, Werner (2003). Context-Awareness on Mobile Devices - the Hydrogen Approach, *Proceedings of the 36th Annual Hawaii International Conference on System Sciences (HICSS'03) - Track 9 - Volume 9*: IEEE Computer Society.

Hull, Richard, Neaves, Philip, & Bedford-Roberts, James (1997). Towards Situated Computing, *Proceedings of the 1st IEEE International Symposium on Wearable Computers*: IEEE Computer Society.

Jones, Rachel (2006). *Experience Models: Where Ethnography and Design Meet.* Paper presented at the EPIC.

Korpip, Panu, Malm, Esko-Juhani, Salminen, Ilkka, Rantakokko, Tapani, & a.o. (2005). Context management for end user development of context-aware applications, *Proceedings of the 6th international conference on Mobile data management*. Ayia Napa, Cyprus: ACM Press.

Kuhn, Thomas (1962). *The Structure of Scientific Revolutions*. Chicago: University of Chicago Press.

Laclau, Ernesto, & Mouffe, Chantal (1985). *Hegemony and Socialist Strategy. Towards a Radical Democratic Politics*. London, New York: Verso.

Meyer, Sven , & Rakotonirainy, Andry (2003). A survey of research on context-aware homes, *Proceedings of the Australasian information security workshop conference on ACSW frontiers 2003 - Volume 21*. Adelaide, Australia: Australian Computer Society, Inc.

Norman, Don (2005). Human-Centered Design Considered Harmful. *Interactions, 12*(4), 14-19.

Nygaard, K, & Bergo, T.O. (1973). *Planlegging, styring og databehandling. Grunnbok for fagbevegelsen (Planning, management and data processing. Handbook for the labour movement)*. Oslo.

Pascoe, Jason (1998). Adding Generic Contextual Capabilities to Wearable Computers, *Proceedings of the 2nd IEEE International Symposium on Wearable Computers*: IEEE Computer Society.

Pateman, Carole (1970). *Participation and Democratic Theory*. Cambridge, London, New York, Melbourne: Cambridge University Press.

Rom, Manuel, Hess, Christopher, Cerqueira, Renato, Ranganathan, Anand, Campbell, Roy, H. , & Nahrstedt, Klara (2002). Gaia: a middleware platform for active spaces. *SIGMOBILE Mob. Comput. Commun. Rev., 6*(4), 65-67.

Schilit, B.N. , Adams, N. , & Want, R. (1994). *Context-Aware Computing Applications.* Paper presented at the 1st International Workshop on Mobile Computing Systems and Applications.

Schütz, Alfred (1962). *Collected papers Vol.I. The problem of social reality*. The Hague: Martinus Nijhoff.

Schütz, Alfred, & Luckmann, Thomas (1974). *The Structures of the Life-World*. London: Heinemann.

Servaes, Jan (1999). *Communication for Development. On World, Multiple Cultures*. Cresskill New Jersey: Hampton Press.

Singh, A. and M. Conway (2006). *Survey of Context aware Frameworks*. Information Technology Services, The University of North Carolina at Chapell Hill.

Sterling, Bruce (2006). Emerging Technology, *Emerging Technology 2006*. San Diego.

Suchman, Lucy (2002). Located accountabilities in technology production. *Scandinavian Journal of Information Systems, 14*(2), 91-105.

Suchman, Lucy (2005). *Introduction to Plans And Situated Actions II: Human-machine reconfigurations, 2nd expanded edition*.

Van Rompaey, Veerle, Van Der Meerssche, Bart, Godon, Marc, Vanden Abeele, Mariek, & Charliers, K. (2005, 25-26 November). *Connecting the family home: Co-designing new technologies for Community Communication.* Paper presented at the ECCR/ECA conference, Amsterdam.

Weiser, M (1991). The computer for the twenty-first century. *Scientific American, Sept.*, 94-104.