Fernando Lyardet · Erwin Aitenbichler · Gerhard Austaller · Jussi Kangasharju · Max Mühlhäuser

Lessons Learned in Smart Environments Engineering

Received: 20.06.2007 / Accepted: 14.07.2007

Abstract During the last few years we have developed different smart environments and artifacts by creating devices or extending existing ones to make them *smart*. In this paper we will show some of the projects developed at the Telekooperation Group. Throughout those examples we show how we have been developing ubiquitous computing aplications by empowering users with new interactions means and extending devices and services to make them smart. We also provide an overview of a middleware and software tools that implement different abstractions required to building a vision were ubiquitous applicacions can be developed in a more efficient and scalable way. Finally, we summarize our experience building smart environments with both top-down and botton-up approaches, presenting the different issues we encountered with the hardware, the software design and development, and the user interaction.

Keywords Smart Products · Ubiquitous Computing · Ambient Intelligence

1 Introduction

During the last 5 years we have developed different ubiquitous computing scenarios and smart environments prototypes. Throughout this process we have gathered several lessons. In particular, when designing and implementing the 'smartness' functionality into new or existing objects rather than the development of the physical device, as for the latter well defined processes and methods already exist.

Building different systems has been very useful to gain an insight of many of the design concerns, and the complexity involved in their integration. With every development we have been shaping the approach of creating smart environments by extending devices and services to make them smart, and transforming them into more powerful and scalable buiding blocks for reifying 'ambient intelligence' functionality.

Another important factor are the users, who are the focus of these applications. Supporting users have been also a subject of our research, in particular how do we develop means to integrate them more seamlessly and effectively with the surrounding technology using different modalities. With this view in mind, we have developed the Talking Assistant, that we present in the following section.

Fernando Lyardet, Erwin Aitenbichler, Gerhard Austaller, Max Mühlhäuser
Telecooperation Group
Darmstadt University of Tecnology
Tel.: +49 6151 16-4267
Fax: +49 6151 16-3052
E-mail: {fernando, erwin, gerhard, jussi, max}@tk.informatik.tu-darmstadt.de

Jussi Kangasharju
Department of Computer Science
University of Helsinki
E-mail: jakangas@cs.helsinki.fi

Fernando Lyardet
also at SAP Research CEC Darmstadt

2 Augmenting Users and Objects

2.1 The Talking Assistant

The Talking Assistant [2] is an audio-based device intended as a general purpose personal device for ubiquitous computing environments. It provides a well-defined minimal functionality which permits very small form factors, allowing the user to carry the device always with her. Audio devices do not need large displays to offer useful interaction possibilities, and through advanced manufacturing techniques, could be reduced to ear-plug size in a few years.

The Talking Assistant is a Pocket PC based system (figure 1) that offers two basic functionalities: Firstly, it integrates a compass to determine the current head orientation of the user. This information is shared with other devices, like the Context Server, via the MundoCore framework. Secondly, it provides a speech recogni-

Fig. 1 The Talking Assistant

tion and synthesis engine, so users can interact with the system via voice.

2.2 Augmenting Devices and Services: Building Smart Products

The notorious absence of large scale smart environments is strongly influenced by the complexity and effort required to build them. Existing examples such as the office [1, 6] or home [4, 12, 20] have been carefully designed top-down by hand at drawing boards, but this approach is fairly limited:

1. In real life there are too many scenarios to be modeled one by one
2. The scenarios, people and technology change over time

Therefore, future ambient intelligent infrastructures must be able to configure themselves from the available, purposeful objects [8] creating small worlds, *"where all kinds of smart devices are continuously working to make inhabitants' lives more comfortable"* [5]. The notion of smarter objects has been also explored in business scenarios for tracking [10, 17], with enhanced sensing and perception to autonomously monitor the physical integrity of goods or to avoid dangerous physical proximity [7]. or to avoid dangerous physical proximity [18]. Together with the increasing ability to embed computing power, also did the capability to network with their surrounding and peers, with ad-hoc interactions with P2P techniques to gather and disseminate information [11, 18]. However, many of these artifacts are based on the ECA (event, condition, action) principle [21]. This first generation of smarter objects, even with increasing embedding complexities are still far from becoming everyday objects people use outside industrial environments, in their homes and offices. Such environments poses new challenges as they are usually managed by people with little or no training, and *"these networks of products are often deployed incrementally, one product at a time"* [13]. A new generation of artifacts called "smart products" is defined by real-world objects, devices or software services bundled with knowledge about themselves and their capabilities, enabling new ways of interacting with humans and the environment autonomously. These properties make Smart Products not only *intelligible* to users, but also *smart* to interpret user's actions and adapt accordingly. By smart we denote the ability of an object to adapt and combine its behavior according to the situation in which it operates that is the set of possible things that are happening and conditions that exist at a particular time and place. We have identified a set of properties we consider are cardinal to Smart products:

- **Self Explanatory.** It refers to the notion that the artifact's operation (be it software or a physical device) must be discoverable without extensive training, from the information provided by the machine itself. It is then self-explanatory *"... to the extent someone looking at the artifact is able to reconstruct the designers intentions regarding its use"* [19]. In computer-enriched artifacts, information can be embedded so that they can explain themselves more like humans do.
- **Self Organizing.** They provide the ability to combine with other federated smart products to provide meaningful, value added functionality. That is, the *"spontaneous formation of well organized structures, patterns, or behaviors, from random initial conditions"* [9].
- **Self Sustainable.** Smart products should be able to work in a self-supported (with ad-hoc connectivity at most) or infrastructure mode (connected to a local network).
- **Extensible.** An important part of a Smart Product's functionality requires some kind of communication capability. Communications allow the SP to be extended and supported locally or by third-party subscriptions.

2.3 A Smart Product Example: The Smart Coffee Machine

The Smart Coffee Machine [3] is a normal off-the-shelf coffee machine from Saeco. When used out of the box, the user can choose between three kinds of coffee namely espresso, small coffee, and large coffee. There is a coffee bean container and a water tank attached to the machine. If either is empty, a small display on the machine prompts to refill them. The display also prompts to empty the coffee grounds container if it is full.

Our modifications allow us to control all the buttons remotely and determine the state of the machine (Figure 2). With this state information and additional RFID readers, we can detect user actions and automatically start processes or proceed in a workflow.

The hardware of the enhanced system consists of the coffee machine, an RFID readers to identify cups, one

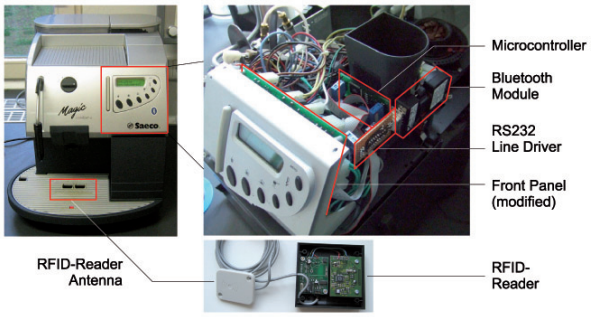

Fig. 2 Hardware components added to the coffee machine

reader for reading the digital keys, and a PC running the control software. The roles of the individual components are as follows.

Coffee Machine: Because the machine does not come with a data interface, we modified the front panel circuit board and attached our own microcontroller to it. This allows us to detect keypresses, simulate keypresses, check if the water tank is empty and read the pump control signal. The latter indicates that the machine is actually dispensing fluid and gives a very accurate measure how much fluid has run through the pump. The machine communicates with the rest of the system via Bluetooth.

RFID reader: The antenna of this reader is attached to the bottom of the coffee machine's cup holder surface. It is positioned such that the reader is able to identify cups as soon as they are placed below the coffee dispensing unit. The RFID tags are glued to the bottom of the cups. This allows the system to start brewing coffee as soon as a cup is put down.

Key reader: Our reader for the digital keys consists of a SimonsVoss Smart Relais, a microcontroller, and a USB interface. Like the electronic locks in our computer science building, the relais can be triggered with the digital keys, which are given to all employees and students. Every user owns a key, and there is a one-to-one relationship between users and keys, which makes these keys highly suitable for identification purposes. In addition, the keys can already serve as simple interaction devices. Because users are required to activate them explicitly by pressing a button, the reception of a key ID can be directly used to trigger actions.

PC: The PC hosts most of the services and is hidden in a cupboard. It gives users voice feedback via the speakers. Users can view their favorite webpages on the monitor. The monitor, keyboard, and mouse are optional and not required to use the core functions of the system.

2.4 The TK Showroom

The combined showroom and meeting room at the Telecooperation Group is equipped with several wall displays, data projectors, tracking systems, smart power plugs, and an audio system. A number of computers are used to drive the displays and host application services. The showroom provides the smart environment in which smart artifacts like the Talking Assistant or the Smart Coffee Machine live in. The Talking Assistant relies on tracking systems in the infrastructure for its location awareness capabilities and the Smart Coffee Machine can benefit from nearby interaction devices. We will discuss the development of smart environments applications in the next section.

3 Lessons Learned in Software Development

Over the past few years we have implemented over 40 applications for smart environments. We have managed to quite successfully solve the networking and distribution issues with our communication middleware MundoCore. To support the development of smart environment applications, we have created a set of common services and development tools.

3.1 Networking Issues

In an environment where networks will be formed spontaneously, hosts must be able to configure themselves automatically and adapt to changing conditions. Furthermore, they must be able to operate in isolation and cannot rely on an infrastructure to be present. From the available networking technologies only a few qualify for spontaneous networking. Related networking issues include the following.

Bluetooth is a PAN technology. Bluetooth was designed as a technology for cable replacement in Personal Area Networks. Because it only supports eight active nodes in a piconet and its slow and power consuming inquiry process, Bluetooth is not good at dynamically adding or removing nodes.

WLAN IP configurations are vastly different. Network properties like IP network class, whether access points permit point-to-point communication between wireless peers, whether access points permit multicast communication, and by which authentication and VPN technology the Internet can be accessed vary vastly between organizations.

Personal firewalls block discovery packets. Personal firewalls virtually inhibit all application-level discovery mechanisms based on IP broadcast or IP multicast communication. Users must first manually configure their firewalls accordingly.

We have put considerable effort into the implementation of efficient mechanisms for spontaneous networking in our communication middleware MundoCore [3]. MundoCore plays an important role as an integration platform. It enables to interface with a large variety of devices, operating systems and vendor-specific APIs in different programming languages.

3.2 Processing of Context Information

Similar to distributed systems platforms, it has been found useful to integrate certain new service classes as common services into ubicomp platforms. While common services in distributed systems platforms are naming, synchronization, caching, object brokering, etc., common services in ubicomp platforms are likely to be location tracking, context processing, task and workflow processing, etc. We will describe some of the services and tools we found useful during the development of multiple different applications in the following.

The Mundo Context Server is responsible for transforming the readings gathered from sensors into information that is meaningful to applications. The function of the Context Server is threefold:

- Interpreting the heterogeneous data received from sensors and transforming this data into a common representation.
- Maintaining a geometric world model of the smart environment.
- Storing histories of sensor data and generated high-level events and supporting queries and subscriptions in those histories.

To derive higher-level context information, the Context Server uses a detailed geometric world model. This model is the virtual counterpart of the real environment which the application targets. It is basically a detailed geometric model of the walls and objects used for visualization purposes. The world model is augmented with a metadata layer that describes objects and regions of interest in space. By means of the world model, the context server is now able to infer higher-level context information from location systems that provide 3D coordinates and orientations, like in which room a user is, which objects are near the user or which object the user is currently looking at. Applications can now send queries or subscribe to the events generated by the Context Server. Thus, the event notifications received by applications are already abstracted from the underlying sensors and describe changes in higher-level context.

3.3 Modeling and Monitoring

WorldView is a tool for creating a spatial model of the smart space (Figure 3). In the main window, the application provides an overview of the available resources. Since many applications are location-sensitive, WorldView provides an easy way to define the regions of interest that should trigger spatial events.

WorldView can be used to inspect the running system by visualizing the events from certain event sources, like tracking systems. If a tag is physically moved around, the position of the corresponding symbol in the map view is updated in real-time. To test applications, WorldView can be used to simulate certain tracking systems. In this case, the user can move around the symbols on the map and WorldView generates the same kind of events the tracking system would.

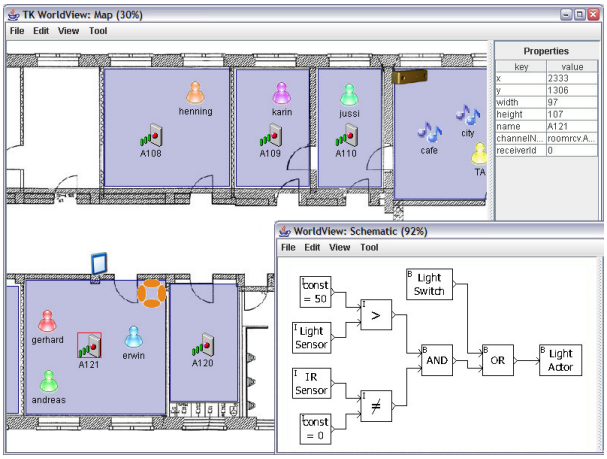

Fig. 3 WorldView map and logic editors

3.4 Application Prototyping

A lot of research in ubiquitous computing is conducted around possible application scenarios that can be put to daily use. Many of these scenarios are quite simple and straightforward to implement. However, many of these applications never come to life, because the whole process from development to deployment is still very complex.

This led us to the idea to create a set of tools to make this development process as easy and fast as possible. An aim was to enable a wide variety of people with some basic technical background, but not necessarily with knowledge of a programming language, to create and alter applications. Simple-structured applications can be directly developed with the provided tools, while more complex applications can be prototyped.

The SYstem for Easy Context Aware Application Development (SYECAAD) [16] facilitates the rapid development of context-aware applications. Applications are built using a graphic-oriented block model. The basic building blocks of applications are sensors, operations, and actors. Operations perform logic or arithmetic operations, implement dictionaries, render HTML pages, etc. Actors can control devices in the smart environment or send feedback to users.

4 Lessons From the Users

We performed different user studies when testing the newly enriched devices. These user studies were performed

as a between subjects study, and they consisted mainly in performance tests designed to gather information through direct observation followed by a debriefing questionnaires. We found a number of aspects that were recurrent in our studies:

Check your assumptions: Understanding the performance impact of adding technology into everyday artifacts challenges our perception. A common assumption with the automation of manual procedures is raw speed: the overall time to accomplish would be lower with the new technology instead of the manual operation. In the case of the smart Saeco, our studies consistently showed that the automated process was slower than the manual operation (time-to-accomplish performance criteria) but the automated behavior was still preferred. On the other hand, the self-explanatory capabilities of the device allowed a dramatic improvement (task-completion rate performance criteria) over following the printed instructions in the user's manual. In the first case, the longer execution times were due to the latencies required by the technology in order to trap, communicate over Bluetooth, process the commands, and trigger back the required operations on the coffee machine. The question of why the automated procedure was still preferred in spite its lower speed led to the next topic.

Attention is a Valuable Resource: Different authors have already identified the user's attention as a scarce resource in desktop applications, and that observation is also true in the ubiquitous computing space: users consistently preferred the automated assistance in spite of proving to be a slightly slower procedure. After interviewing the users to collect their subjective impressions, we found out that smart functionality was preferred because it requires lower or no user attention to trigger and control the device's operation.

Multimodality Challenges: The ability to communicate through different channels adds a number of problems previously unknown in desktop applications. We have studied the design of multimodal and voice user interfaces in the context of eyes-free, hands free scenarios such as blue collar workers [14], and consumer devices [15], but the pervasive integration of voice interfaces is a strong commitment to capturing the user's attention: while it is possible to avoid watching a display, it is impossible to stop listening. Visual feedback can be personal while aural feedback is always collective. These examples show some of the concerns involved when integrating new modalities: consistency, orientation, cognitive overload and privacy.

Ultimately, we must reflect on how the integration of modalities such as voice contribute to a more flexible interaction or rather they become a new kind of informative pollution. An interesting example appears in the case of the Talking Assistant, where the user is wearing a special headset working usually in noisy industrial environments and is able to navigate through menues and documents using voice activated commands. In this case, the feedback is only heard by the user, who can control the application and navigate through documents using single word commands, and the user's voice impact is mitigated by the environmental noise.

The ability of being able to interact both ways in certain modalities such as audio is important but still possible only in a few cases. Our experience with the current state of the art technology tells us that the use of voice interaction must be also be balanced against a highly asymmetric interaction, and the resulting danger of cross multimodality confusion. Most of current voice user interface technologies allow a minimal interaction with the user: while the device can provide spoken feedback and explanations through Text To Speech (TTS), often users cannot ask back, acknowledge or cancel an operation, and they must resort to other input modalities.

Ability to Recognize, Remember and Adapt: In the examples we presented, we have applied different mechanisms to recognize objects and identify the users. Some of these technologies can be used for both purposes, but as we have also seen in the preceding examples, they can be different as well. Sometimes, technology adoption is defined by what technologies are already deployed and adopted by the users, for instance in the case of using the smart keys as a user identification mechanism instead of rfid.

At least one of these technologies is required for any smart product, for instance enriching cups with rfid tags allows recognizing the object type and its properties. However, not always both technologies need to be present: we can allow the association of one entity with other (e.g., a cup with a person) to infer the information we require to retrieve or "remember" the knowledge of previous experiences (e.g., user preferences), and apply this knowledge to adapt the behavior, that is, reacting in a "smart" way.

The decision of what kinds of associations we introduce has practical implications: adding an extra identification mechanism adds knowledge and certainty, but it also means adding an extra step to the user interaction process. We tested in our smart coffee machine scenario two different approaches:

1. Enforcing user identity: every time users had to use their electronic key to trigger the automatic features
2. Temporal associations: users identify themselves to temporarily associate themselves with an object, or renew a previous association. After a period of time, associations would disappear.

The decision proved to be task based: temporal associations were more appropiate for the most frequent tasks, with periodical identification enforcement. In the coffee machine example, users would need to identify themselves with their smart keys to associate a cup with them, but afterwards, only the cup was necessary for the system to identify the user preferences.

5 Conclusions

The examples we presented in this paper show that a succesful design of smart environments involves the clear definition of the usage scenarios, actors, objects and activities as a first rationale to determine what functionality and adaptation schemes will be required. We found this first analysis critical to avoid costly pitfalls, regardless of whether the development will follow a top-down, or bottom-up approach based on the gradual addition of smart objects. These approaches do define different assumptions on the specialization of the functionality, scalability and ease of deployment. We believe both approaches will co-exist, but for different scenarios.

Finally, future work should seek the development of design and development artifacts to better identify what are the specific design and implementation concerns, and isolate the different technological components, a task that today is still very much unstructured.

References

1. M. Addlesee, R. Curwen, S. Hodges, J. Newman, P. Steggles, and A. Hopper A. Ward. Implementing a sentient computing system. In *IEEE Computer*, volume 34, pages 50–56. IEEE, August 2001.
2. Erwin Aitenbichler, Jussi Kangasharju, and Max Mühlhäuser. Talking Assistant: A Smart Digital Identity for Ubiquitous Computing. In *Advances in Pervasive Computing*, pages 279–284. Austrian Computer Society (OCG), Austrian Computer Society (OCG), 2004.
3. Erwin Aitenbichler, Jussi Kangasharju, and Max Mühlhäuser. MundoCore: A Light-weight Infrastructure for Pervasive Computing. *Pervasive and Mobile Computing*, pages 332–361, 2007. doi:10.1016/j.pmcj.2007.04.002.
4. B. Brumitt, B. Meyers, J. Krumm, A. Kern, , and S. Shafer. Easyliving: Technologies for intelligent environments. In *Proc. of 2nd International Symposium on Handheld and Ubiquitous Computing (HUC 2000)*, pages 12–27. Springer, September 2000.
5. Diane Cook and Sajal Das. *Smart Environments: Technology, Protocols and Applications (Wiley Series on Parallel and Distributed Computing)*. Wiley-Interscience, 2004.
6. J.R. Cooperstock, S. S. Fels, W. Buxton, and K. C. Smith. Reactive environments: Throwing away your keyboard and mouse. In *Comm of the ACM*, volume 40, pages 50–56. IEEE, September 1997.
7. C. Decker, M. Beigl, A. Krohn, P. Robinson, and U. Kubach. eseal - a system for enhanced electronic assertion of authenticity and integrity. In *Proc. of Pervasive 2004*, pages 18–32. Springer, April 2004.
8. J.L. Encarnacao and T. Kirste. *Ambient Intelligence: Towards Smart Appliance En-sembles*, volume LNCS 3379, pages 261–270. Springer, 2005.
9. J. Rocha Jr. et al. X-peer: A middleware for peer-to-peer applications. In *Proceedings of 1st Brazilian Wksp. Peer-to-Peer*, 2005.
10. A. Fano and A. Gershman. The future of business services in the age of ubiquitous computing. In *Comm of the ACM*, volume 45, pages 83–87. ACM Press, December 2002.
11. Andreas Heinamann and Max Mühlhäuser. *Spontaneous Collaboration in Mobile Peer-to-Peer Networks*, volume LNCS 3485, chapter 25, pages 419–433. Springer, 2005.
12. C. Kidd, R. Orr, G. Abowd, C. Atkeson, I. Essa, B. MacIntyre, E. Mynatt, T. Starner, and W. Newstetter. The aware home: A living laboratory for ubiquitous computing research. In *Proc. of the Second International Workshop on Cooperative Buildings (CoBuild99)*, October 1999.
13. F. Reynolds. The ubiquitous web, upnp and smart homes. *Pervasive Computing Group, Nokia Research Center, Cambridge*, 2004.
14. Dirk Schnelle, Erwin Aitenbichler, Jussi Kangasharju, and Max Mühlhäuser. Talking assistant - car repair shop demo. http://elara.tk.informatik.tu-darmstadt.de/publications/2004/taCarRepairShop.pdf, 2004.
15. Dirk Schnelle and Fernando Lyardet. Consumer Voice User Interface Design Patterns. In *Proceedings of 12th European Conference on Pattern Languages of Programs (EuroPlop 2007)*, 2007. to appear.
16. Jean Schütz. SYECAAD: Ein System zur einfachen Erzeugung kontextsensitiver Applikationen. Master's thesis, Technische Universität Darmstadt, 2005.
17. F. Siegemund and C. Flrkemeier. Interaction in pervasive computing settings using bluetooth-enabled active tags and passive rfid technology together with mobile phones. In *Proc. IEEE PerCom 2003*, pages 50–57. Springer, March 2003.
18. Martin Strohbach, Hans-Werner Gellersen, Gerd Kortuem, and Christian Kray. Cooperative artefacts: Assessing real world situations with embedded technology. In *Ubicomp*, pages 250–267, 2004.
19. Lucy A. Suchman. *Human-Machine Reconfigurations: Plans and Situated Actions*. Cambridge University Press, New York, NY, USA, 2006.
20. E. M. Tapia, S. Intille, and K. Larson. Activity recognition in the home using simple and ubiquitous sensors. In *Proc. of Pervasive 2004*, pages 158 – 175. Springer, April 2004.
21. Tsutomu Terada, Masahiko Tsukamoto, Keisuke Hayakawa, Tomoki Yoshihisa, Yasue Kishino, Atsushi Kashitani, and Shojiro Nishio. Ubiquitous chip: A rule-based i/o control device for ubiquitous computing. In *Pervasive*, pages 238–253, 2004.