MIDDLEWARE FOR BUILDING PERVASIVE SYSTEMS

Amro Al-Akkad, Ferry Pramudianto, Marco Jahn, Andreas Zimmermann
Fraunhofer Institute for Applied Information Technology
Schloss Birlinghoven, 53754 Sankt Augustin, Germany.

{amro.al-akkad, ferry.pramudianto, marco.jahn, andreas.zimmermann}@fit.fraunhofer.de

ABSTRACT

Pervasive software succeeds when it is leveraged permeable into the background of everyday life and work supporting users more properly to fulfill their tasks than they used to do it with pre-existing techniques. Hence, we hypothesize, that a particular pervasive system may comprise certain styles of intelligent networking embedded systems, secure communication patterns, context-aware assisting, novel techniques of interaction and many more sophisticated features. Building such highly flexible systems poses diverse challenges for an underlying middleware platform. This paper reports—from a developers' point of view—on experiences we gained from using a generic middleware framework that targets the rapid development of pervasive systems. Based on two case studies, conducted in the healthcare and home automation domain, we describe in detail how we designed and deployed two prototypical ubiquitous applications. We elaborate the difficulties our designated middleware framework helped to us to overcome, and at the same time look on the intrinsic complexity networked embedded systems constitute. Finally, we conclude with a set of lessons learned for designing pervasive systems and sketch ideas for future work.

KEYWORDS

Pervasive & Ubiquitous Computing, Middleware, Embedded Networked Systems and Service-Oriented Architecture.

1. INTRODUCTION

In the recent years, we have seen how technology makes more and more step forwards towards Weiser's vision (Weiser September, 1991) of pervasive computing. Today, continuously increased computing power is embedded into smaller devices. Common objects are enhanced with sensors and actuators shaping new 'smart objects'. This drives the development of pervasive computing systems that assists users to focus on their actual tasks without having to deal with the technological complexity. Reaching such stage of high flexibility poses many challenges. (Satyanarayanan 2001) has described in detail diverse challenges the emerging field of pervasive computing may need to deal with. Further on, (Want and Pering 2005) reports: where pervasive systems have been successfully deployed, where these systems still have limitations, and where big challenges remain.

This paper describes—from a developers' point of view—experiences we gained from using a dedicated middleware framework that targets the rapid development of pervasive systems in diverse domains. Based on two case studies, conducted in the healthcare and home automation domain, we describe in detail how we designed and deployed two prototypical ubiquitous applications. We elaborate the difficulties our designated middleware framework helped to us to overcome, and at the same time look on the intrinsic complexity networked embedded systems constitute. We were motivated to do research of pervasive computing in the home automation and healthcare domain, as previous studies like (Estrin, Culler et al. 2002) have shown that there might be a great benefit for both domains, and like to motivate both application domains with examples where ubiquitous techniques have already been applied successfully.

Home Automation: While embedding sensors and actuators into common devices installed at home, these devices will become 'smart objects' being able to discover other devices and communicate with them within an intelligent sensor network building the common ground for 'smart home' environments. Home automation is supposed to cover a wide range of ubiquitous applications assisting users proactively in doing their daily

activities, adapting unobtrusively to the needs of inhabitants in their living environment, to provide a higher standard of comfort and security, and to maintain energy savings (Tränkler and Schneider 2001; Augusto and Nugent 2006). 'Smart home' environments are still evolving; the first one was the TRON-house[1] built within a pilot-project in Tokyo in 1989, which contained 360 integrated computers. Philips runs the HomeLab[2] project platform in Eindhoven to do research in home automation, networking of embedded devices, and multimedia entertainment: smart mirrors or windows are capable of displaying video streams. The 'smart carpet'[3] has been designed, which arranges into the carpeting a tessellated pattern of microchips and RFID sensors being able to detect persons through pressure and to navigate intelligently servant robots as well.

eHealth: The healthcare domain constitutes a wide field of application for pervasive computing. Enhancing this area with ubiquitous systems can improve our quality of life. For instance, 'smart objects' installed at the patient's home can serve the medical assessment of patients requiring permanent care, and thus help to create a prevention for late major complications which may endanger life. Furthermore, an intelligent P2P network of embedded systems may facilitate secure communication of data, e.g. vital values, patient data history, medical instructions etc, among doctors, physicians, and patients. Although, pervasive computing in the healthcare domain is still in its beginnings, in the recent years some ambient, semi-automatic applications have been released. For example, SIMPILL[4] company has designed an intelligent pill box that monitors the medication intake and reminds the patient in real-time, if he forgets to take the medication as prescribed. VivoMetrics[5] has developed a 'life shirt' that supports the ambient monitoring of more than 30 vital values as the blood pressure, heart rate, or oxygen consumption. Within a pilot project summarized in (Intel 2005), patients are equipped with a wristband that incorporates an RFID chip allowing mobile identification of patients and quick access of patient data as well.

2. RELATED WORK

Some middleware frameworks have been developed to support the development of pervasive systems. A general approach was addressed by SOCRADES (Karnouskos, Baecker et al. 2007.; Kirkham, Savio et al. 2008), in which a Service Oriented Architecture (SOA) based framework for embedded systems was developed. It includes many features, e.g. service discovery, composition, and management. In another project called AMIGO (Ressel, Ziegler et al. 2006) there were attempts to establish an open standard for networked home environments.

The study of (Raatikainen, Christensen et al. 2002) examines requirements for future middleware in order to support properly pervasive applications in the healthcare and home entertainment domain. In (Bardram and Christensen 2001) a middleware architecture for pervasive healthcare is described that supports activities in hospitals, such as the prescription or handout of medicine. The paper of (Kunze, Grossmann et al. 2002) describes an ubiquitous application that perceives to monitor the patient's vital values without constraining his mobility; in this setting patients transmit their vital values to a medical base station through sensors being weaved into textiles. Based on a survey (Kumar, Kambhatla et al. 2008) give an overview of current and future techniques considering specific quality-of-service parameters to support a remote monitoring of patients' cardiac condition.

Developing intelligent pervasive systems for home environments involves a highly dynamically complex orchestration of different software and hardware components. Nowadays, several home automation frameworks that strive to encapsulate this complexity exist. Bodhuin (Bodhuin, Canfora et al. 2006) discussed a software architecture that abstracts the representation of physical devices facilitating the connection to the virtual world in order to hide the complexity and heterogeneity that exists in the physical

[1] Sakamura Laboratory. TRON Intelligent House. Available from: http://tronweb.super-nova.co.jp/tronintlhouse.html.

[2] HomeLab. 2002; Available from: http://www.research.philips.com/newscenter/pictures/misc-homelab.html.

[3] "Vorwerk is presenting the first carpet containing integrated RFID technology for the intelligent navigation of service robots. An outstanding lead in technology resulting from a surface coverage and control accuracy unattained until now." Retrieved 27 July, 2009, from http://www.vorwerk-teppich.de/sc/vorwerk/rfid_en.html

[4] SIMpill (copyright SIMpill © 2008). Available from: http://www.simpill.com.

[5] VIVOMETRICS, lifeshirt sytem. 2004; Available from: http://www.vivometrics.com/

world. Heterogeneous devices constitute different software libraries and component technology which is hard to maintain. Therefore, HAAIS-DSL (Clemente, Conejero et al. 2009) suggested a rather generic architectural approach based on a Model Driven Development that abstracts the model and decouples it from the implementation technology, such as programming languages and hardware components. A SOA for devices (Jammes, Mensch et al. 2005) constitutes a way to encapsulate a low-level device communication and allows devices to be integrated seamlessly into processes, and thus provides the flexibility and dynamic changes pervasive environments may require (Jahn, Pramudianto et al. 2009).
Though the costs for microcontrollers, sensors, and actuators have become less in the recent years, they are not yet commonly adopted for common use, as currently the development of reusable software for instantiating pervasive environments is not available (Norbisrath and Mosler 2006). Therefore, there is a need for a generic software system approach in order to allow for mass productions that may suppress the development costs. SOCRADES and AMIGO are examples for rather generic middleware frameworks, though we preferred to use Hydra[6] representing an enhanced middleware framework that covers eHealth and home automation as application domains to explore user experiences in regard of pervasive environments.

3. FRAMEWORK

"A framework is a reusable design that requires software components to function." (Murray, Carrington et al. 2004). The Hydra middleware framework comprises a set of reusable service components, and various visual tools such as a device and service catalogue, to support both device developers and application developers in interconnecting heterogeneous embedded systems through an overlay P2P network (Milagro, Antolin et al. June 2009). Hydra is built on a SOA, which relies on Web Services as communication channels among components. The services mostly are developed as bundles conform to the OSGi[7] specification, as it provides a dynamic and secure module system for extensible Java modules for the easy and quick deployment. Hydra separates the development into application and device, in order to decouple the application logic from the device communication logic. Figure 1 gives an overview of the Hydra middleware architecture.

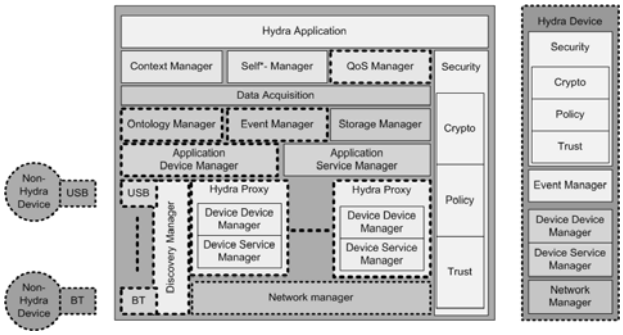

Figure 1. HYDRA architecture (components that were used are marked in dot lines).

In the following we will describe the middleware components and tools we used to build our two prototypes:

Middleware Components

- **Network Manager:** The Network Manager is the entry and exit point of information flowing in Hydra network. It represents a sophisticated middleware software component. While applying the SOAP tunneling pattern (Milagro, Antolin et al. September 2008) it facilitates the communication among services beyond firewalls and NATs. Further, it establishes a custom-made mobile IP scheme to deal with pervasive services, and thus solves the thrust in pervasive systems when endpoints suddenly disappear or reoccur as well due to changing locations (mobility) or network connection errors (Satyanarayanan 2001).
- **Security:** Hydra Middleware incorporates from the outset in its design high-level security, trust and privacy concerns (Hoffmann, Badii et al. 2008). For instance, security issues as the loss of control, profiling and surveillance are considered accordingly.

[6] European Research Project HYDRA IP (IST-2005-034891), http://www.hydramiddleware.eu.
[7] OSGi Service Platform. Core Specification, Release 4, Version 4.1. 2007: OSGi Alliance.

- **Event Manager:** The nature of Web Services is that they only provide routines that can be called, though they cannot proactively initiate the communication. On the other hand, when dealing with devices, sometimes they need to proactively initiate the communication upon state changes in the devices. Therefore, Hydra defines a mechanism so that a device can publish an event, and the corresponding application can subscribe for this event and get notified when the event is published.
- **Quality-of-Service Manager:** The Quality-of-Service (QoS) Manager is a service component that applies a semantic service selection algorithm to support the decision for selecting best-suitable services out of a range of services with same functionality, though differing in respect to non-functional service properties as availability, efficiency, or power consumption.

Tools

- **Limbo:** Limbo is an efficient compiler making possible to deploy minimal Web Service code on resource-constrained devices. It addresses several challenges of running successfully Web Services on embedded devices, such as the efficiency and performance of Web Services, the dependencies of hardware and software platforms, changes of the state and context of an embedded device (Hansen, Zhang et al. 2008a; Hansen, Zhang et al. 2008b). To make Limbo aware of its compilation context, an ontology set compliant to the Web Ontology Language (OWL[8]) is used.
- **Discovery:** Hydra supports the device discovery regardless of their heterogeneous network protocols such as Bluetooth, RF, and Zigbee. It generalizes the discovery process of various protocols by generating an UPnP proxy for other protocols, so that these devices can be discovered on the IP layer.

4. DESIGN & DEPLOYMENT

This chapter describes the two prototypical ubiquitous applications we have built on Hydra middleware. For each, we first tell the story, outline the architecture design, and close with experiences and lessons learned.

4.1. eHealth Test Bed

Scenario: Ruby is a 65 year old retiree that has a chronic heart disease that needs regular check-ups. Since her only son was sent to work far away, she is not getting those check-ups as prescribed, as it is quite hard for her to go on her own to the hospital. Therefore her health insurance company recommended her that she should try an eHealth system that allows her health information be monitored remotely by her physician. Now, every morning, she has to weigh herself on a scale, take her body temperature, and wear an ECG belt on her chest. Her health information is then sent through a secured channel in the internet. One morning, she felt sick, and her ECG indicated that there was something wrong. Her physician sends her a message and told her to take her temperature and weight. When he received her critical values, he decides to bring her into a hospital for further examination. He informed her that he is sending a nursing service. Ruby grants the nursing service an access to her house so that they can directly enter and pick her up from her bed. The nursing station is informed with her current health state, a navigational map, and an access token that was generated from Ruby's home system. The nurses prepare themselves according to the information. After arriving at Ruby's house, they authenticate themselves using a RFID card granting them secure access to the patient's home. The home system recognizes the security token on the card and unlocks the door. They get in and pick Ruby up from her bed and bring her to the hospital.

Design: Figure 2 depicts our eHealth architecture. In the patient's home, we interconnected various sensors communicating over different protocols such as USB for the RFID Reader, Bluetooth for the scale and ECG, serial port for the thermometer, and WiFi for the smart phone. We overcame this heterogeneity by using a middleware that encapsulates the lower network protocol and only exposes web service interfaces to the application. The server detects the available sensors in the environment and creates a web service proxy for each of them. The application on the terminal and smart phone then sends a semantic query of services that it needs to the service registry on the server, which is then replied with the service addresses. The patient can

[8] OWL Web Ontology Language Guide. 2004; Available from: http://www.w3.org/TR/owl-guide/

see his health information history, communicate with his physician and initiate a measurement from his touch screen enabled phone. The eHealth system is also connected to the patient's home security system. In case of an emergency, a care taking service is only allowed to enter the patient's home and pick him up after authenticating himself by the use of a RFID card reader. The sensory information is further transferred to the patient's terminal and mobile phone, but also to the hospital terminal where his physician can see medical values and alert when there is any abnormal information. The data transfer to the hospital goes over a secured P2P network.

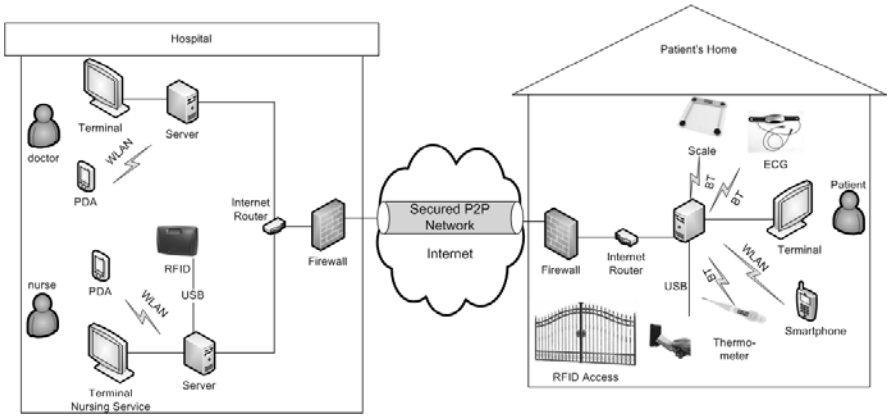

Figure 2. ARCHITECTURE of eHealth test bed

Experiences and Lessons Learned: The Hydra Network Manager can be deployed on devices with sufficient resources, e.g. smart phones as the Android G1[9], PlayStation 3[10] (PS3), or on resourceful devices that are able to host proxies for resource-constrained devices. When we tried to deploy the Hydra Network Manager on the PS3 running Ubuntu 8.04, too less main memory was available to run it. Therefore, we had to replace the operating system with Helios, a simplified Yellow Dog Linux dedicated for the PS3, which turns out working well with the Network Manager.

Hydra offers device developers a rich collection of tools and libraries that enable them to build a Hydra device from scratch as well as building proxies for existing non-Hydra devices. When developing a Hydra device or a proxy, developers have to define web services and UPnP stacks. The latter is used for the discovery process. Currently, Hydra provides developers two development platforms: .NET and Java. Referring to .NET, Hydra offers generic libraries that include various communication protocols such as Bluetooth and USB. For Java developers, Hydra offers the Limbo compiler that generates the web service stack as well as UPnP protocol stack from WSDL[11] and a semantic device description in OWL[12]. Then, the semantic description is saved in Hydra ontology, so that applications can semantically query devices. For developing proxies, the Hydra framework clearly separates the logic that communicates with legacy devices in a so called *Device Service Manager* (outside Hydra network) and the logic providing Web Service interfaces to be consumed within the Hydra network in a layer called *Device Device Manager*.

Hydra offers application developers an interface through an *Application Device Manager* and an *Application Service Manager*. The former gives developers information about available devices within the Hydra network and a way to filter this information based on the device location and type, meanwhile the latter supplies developers with semantic filtering and composition of services.

4.2. Home Automation Test Bed

Scenario: Lucy is doing her masters in computer science. She passed the second semester, and recently decided to study courses with special emphasis on ubiquitous interactive systems. That is why she has recently purchased a new cell phone equipped with some fancy functionality, such as projecting a video

[9] Android G1, http://www.androidg1.org/

[10] PlayStation 3, http://www.us.playstation.com/, © 2009 Sony Computer Entertainment America Inc. All Rights Reserved

[11] Web Service Description Language (http://www.w3.org/TR/wsdl)

[12] Web Ontology Language (http://www.w3.org/TR/owl-features/)

stream to another device capable of playing back a video. Now Lucy is walking through the streets on her way to her classes, and sees a poster advertising the new Harry Potter movie. The poster consists of two action fields that Lucy can interact with while using her mobile device. First field provides the download of a video trailer, and second one directs you to the booking page for buying tickets. As Lucy likes to watch the video, she scans the first action field, and after the trailer is downloaded, it is automatically played back on her cell phone. Unfortunately, Lucy does not enjoy watching the video on her mobile, as the resolution of her cell phone is too low. Kevin, a PhD Student passes by and sees sad Lucy. As he knows Lucy is a big Harry Potter fan, he suggests her to go to the home automation lab at the university to watch there the trailer on a more appropriate playback device. The home automation lab is equipped with a Hi-fi, LCD screen, and a data projector. Arriving at the lab they ask an application installed on the cell phone to scan the environment for 'PlayVideo' services. After some short while the mobile signals them that the data projector is the best-suitable device for playing the video, as the LCD screen does not have any speaker attached. While carrying her mobile with an intuitive gesture 'take it' Lucy points at the projector screen, and the trailer is played back on it. Lucy enjoys watching the video, and invites Kevin for a coffee in return. In the evening both meet for a date, and...

Design: Our scenario is inspired by work of (Henze, Rukzio et al. 2008), which describes the association between physical and virtual world and that a physical object may offer virtual services which can be bookmarked by users. Figure 2 shows that the user takes a picture of a physical poster on the street, and then he sends it to an image recognition server via internet. The server that holds the association replies him with information about services, such as shops to buy tickets, or to download a video trailer.

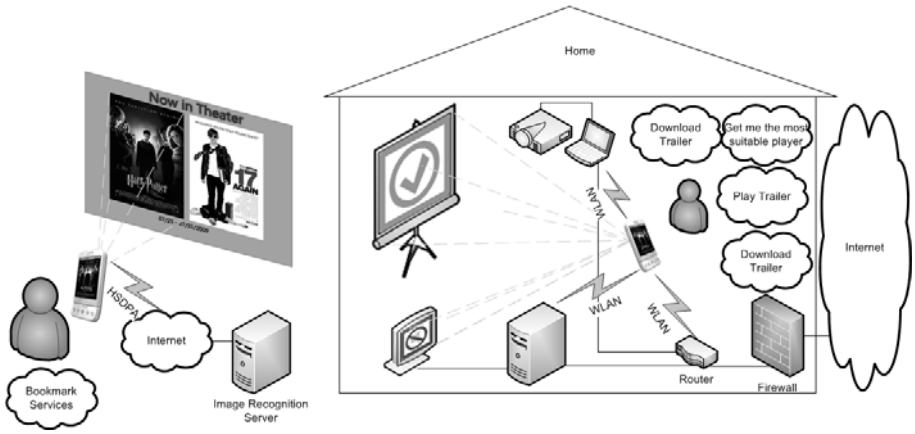

Figure 3. ARCHITECTURE of home automation test bed

The mobile phone bookmarks this services information locally, which later can be accessed by the user. For example, latter on she accesses the download-trailer service, and has the video clip saved in her phone, though she wants to play it on a larger screen. Since all services are annotated with *service requirements* and all devices are annotated with *device capabilities*, the Hydra QoS Manager is able to calculate the devices' suitability to the user's preference. So she scans the room with her projector phone for the best-suitable display. The projector phone beams the information onto the device where the user is pointing at to show how suitable the device is for playing the trailer. The devices in the environment are tagged with infrared LEDs, so that the cell phone is able to detect which device it currently points at. The user chooses the best matched screen, the movie is streamed into the corresponding computer, and the movie is played.

Experiences and Lessons Learned: We developed our mobile application on the G1 cell phone using Android 1.5 SDK. As Android uses other implementation of Java than SUN Java-derivative, we experienced that we needed specific libraries for Android. For instance, it does not support common Java Web Service libraries, such as Axis[13] and JAX-WS [14]. Hence, we used kSOAP2[15], a lightweight Web Service API for

[13] "Apache Axis is an implementation of the SOAP ("Simple Object Access Protocol") submission to W3C", http://ws.apache.org/axis/

[14] Java API for XML Web Services (https://jax-ws.dev.java.net/)

[15] kSOAP2, http://ksoap2.sourceforge.net/

constrained Java environments. Therefore, we had to write client stubs in kSOAP2 manually in order to perform Web Service calls instead of using common Web Service libraries to generate client-side stubs.
For supporting interoperability with other Web Service frameworks as .NET, we preferred to skip the use of complex types as parameters of the Web Service interfaces, and decided to use XML in String data types. For building and parsing the parameters we used kXML2 library that was customized for Android.
In order to make Hydra Web Service calls from the phone, we deployed on it a lightweight Hydra Network Manager version that has been specifically developed for Android. This porting process on Android was simplified by using the Limbo compiler. In general, we succeeded in porting and consuming parts of Hydra Middleware, although this required some tweaks due to the quirk of Android's virtual machine called Dalvik.
Moreover, we used the QoS Manager that proves the concept of semantic service selection. It helped us to support the user in our scenario to make the decision of playing the video on the best-suitable device. Currently, the QoS Manager supports two ways of service selection. First, it considers for each non-functional service property how it matches the needs of the users, and then calculates the average for all matching rates. Second, it considers an extreme criterion for service selection, such as taking care of the device with the highest resolution, although another device yields a higher average matching rate.

5. CONCLUSION & FUTURE WORK

In this paper we have reported on experiences and lessons learned we gained from using a generic, pervasive middleware framework for building two prototypical ubiquitous applications. Hydra middleware helped us to overcome the complexity of wiring heterogeneous embedded devices to an intelligent network that assists users in their daily activities. After getting acquainted with the main architectural parts of the middleware framework, we have been able to build "out-of-the-box" two pervasive prototypes while focusing on the essentials of our applications.
By our two prototypical applications we have given examples how beneficial pervasive computing can be in the eHealth and home automation domain. Further, we think both domains are complementary to a certain degree: an intelligent network of smarter objects at the patient's home offers elder, sick or disabled people an opportunity to live at home, and at the same time shorten hospitalization periods. Therefore, on the basis of other applications and our two case studies, we like to motivate for further research of pervasive computing in healthcare and 'smart home' environments.
Pervasive software succeeds when it is leveraged permeable into the background of everyday life and work supporting users more properly to fulfill their tasks than they used to do it with pre-existing techniques. Although, we might have good knowledge of technology, we currently cannot really estimate how pervasive computing affects social aspects and needs in future society. It is nearly impossible to build fail-safe applications. Hence, we would like to view pervasive environment from a critical angle referring to system failures and proper switchover facilities. We think approaching this view may considerably help to improve the functional security in pervasive computing settings.
As future work, we plan to conduct ethnographical studies of our two prototypical applications in order to gain more results concerning the appropriateness of our pervasive scenarios. These results may help us to better inform the design of our applications, and in turn allow us to update the requirements for an underlying pervasive middleware.

ACKNOWLEDGEMENTS

This work is partially funded by the European research project Hydra IP (IST-2005-034891). We like to thank our colleagues at Fraunhofer FIT and our partners in the Hydra project for their support.

REFERENCES

Augusto, J. C. and C. D. Nugent (2006). Designing Smart Homes: The Role of Artificial Intelligence, Springer.

Bardram, J. E. and H. B. Christensen (2001). Middleware for Pervasive Healthcare. In Advanced TopicWorkshop: Middleware for Mobile Computing. . IFIP/ACM Middleware 2001 Conference.

Bodhuin, T., G. Canfora, et al. (2006). Hiding complexity and heterogeneity of the physical world in smart living environments. Proceedings of the 2006 ACM symposium on Applied computing. Dijon, France, ACM.

Clemente, P. J., J. M. Conejero, et al. (2009). HAAIS-DSL: DSL to develop home automation and ambient intelligence systems. Proceedings of the Second Workshop on Isolation and Integration in Embedded Systems. Nuremburg, Germany, ACM.

Estrin, D., D. Culler, et al. (2002). "Connecting the Physical World with Pervasive Networks." IEEE Pervasive Computing **1**(1): 59-69.

Hansen, K. M., W. Zhang, et al. (2008a). Flexible Generation of Pervasive Web Services Using OSGi Declarative Services and OWL Ontologies. Proceedings of the 2008 15th Asia-Pacific Software Engineering Conference, IEEE Computer Society.

Hansen, K. M., W. Zhang, et al. (2008b). Limbo: an ontology based web service compiler for networked embedded devices. International Conference on Software Reuse, Beijing China.

Henze, N., E. Rukzio, et al. (2008). Physical-virtual linkage with contextual bookmarks. Proceedings of the 10th international conference on Human computer interaction with mobile devices and services. Amsterdam, The Netherlands, ACM.

Hoffmann, M., A. Badii, et al. (2008). Towards Semantic Resolution of Security in Ambient Environments Developing Ambient Intelligence, Proceedings of the International Conference on Ambient Intelligence Developments (AmI.d'07), Springer Paris. **1:** 13-22.

Intel. (2005). "RFID erstmalig in deutschem Krankenhaus: Intel, Siemens Business Services und Fujitsu Siemens Computers starten am Klinikum Saarbrücken RFID-Pilotprojekt." Retrieved 27 July, 2009, from http://www.intel.com/cd/corporate/pressroom/emea/deu/212727.htm.

Jahn, M., F. Pramudianto, et al. (2009). Hydra Middleware for Developing Pervasive Systems: A Case Study in the e-Health Domain. Proceedings of the 1st International Workshop on Distributed Computing in Ambient Environments (DiComAe). Paderborn.

Jammes, F., A. Mensch, et al. (2005). Service-oriented device communications using the devices profile for web services. Proceedings of the 3rd international workshop on Middleware for pervasive and ad-hoc computing. Grenoble, France, ACM.

Karnouskos, S., O. Baecker, et al. (2007.). Integration of SOA-ready networked embedded devices in enterprise systems via a cross-layered web service infrastructure. Emerging Technologies & Factory Automation. E. IEEE and C. on**:** 293-300.

Kirkham, T., D. Savio, et al. (2008). SOA middleware and automation: Services, applications and architectures. Industrial Informatics INDIN 2008, 6th IEEE International Conference on 1419-1424.

Kumar, S., K. Kambhatla, et al. (2008). "Ubiquitous computing for remote cardiac patient monitoring: a survey." Int. J. Telemedicine Appl. **2008**: 1-19.

Kunze, C., U. Grossmann, et al. (2002). "Application of Ubiquitous Computing in Personal Health Monitoring Systems." In Biomedizinische Technik **47** (Beiträge zur 36. Jahrestagung der Deutschen Gesellschaft fuer Biomedizinische Technik): 3.

Milagro, F., P. Antolin, et al. (June 2009). Pervasive Web Services over P2P Network. Smart Homes Infrastructure and Interactions. University of Groningen, The Netherlands, IEEE WETICE 2009.

Milagro, F., P. Antolin, et al. (September 2008). Pervasive Service on P2P Network, ARCHITECTURAL ISSUES AND TOOLS FOR THE IOTS. The Internet of Things and Services 1st International Research Workshop. Sophia-Antipolis, French Riviera.

Murray, L., D. Carrington, et al. (2004). An approach to specifying software frameworks. Proceedings of the 27th Australasian conference on Computer science - Volume 26. Dunedin, New Zealand, Australian Computer Society, Inc.

Norbisrath, U. and C. Mosler (2006). Functionality configuration for eHome systems. Proceedings of the 2006 conference of the Center for Advanced Studies on Collaborative research. Toronto, Ontario, Canada, ACM.

Raatikainen, K., H. B. Christensen, et al. (2002). "Application requirements for middleware for mobile and pervasive systems." SIGMOBILE Mob. Comput. Commun. Rev. **6**(4): 16-24.

Ressel, C., J. Ziegler, et al. (2006). An approach towards personalized user interfaces for ambient intelligent home environments. Intelligent Environments, IE 06. 2nd IET International Conference on 247-255.

Satyanarayanan, M. (2001). "Pervasive Computing: Vision and Challenges." Personal Communications, IEEE **8**(4): 8.

Tränkler, H.-R. and F. Schneider (2001). Das intelligente Haus (Arbeiten und Wohnen mit zukunftsweisender Technik), Richard Pflaum Verlag, München.

Want, R. and T. Pering (2005). System challenges for ubiquitous \& pervasive computing. Proceedings of the 27th international conference on Software engineering. St. Louis, MO, USA, ACM.

Weiser, M. (September, 1991). "The computer for the 21st century." Scientific American **3**: 10.