The 18th Annual IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC'07)

CREATING NEXT GENERATION APPLICATIONS AND SERVICES FOR MOBILE DEVICES: CHALLENGES AND OPPORTUNITIES

Agathe Battestini
agathe.battestini@nokia.com

Christian Del Rosso
christian.del-rosso@nokia.com

Adrian Flanagan
adrian.flanagan@nokia.com

Markus Miettinen
markus.miettinen@nokia.com

Nokia Research Center
PO Box 407, 00045
Helsinki, Finland

ABSTRACT

Next generation applications and services have to be user-centric, and therefore must satisfy users' needs and improve users' lives. These systems also have to be able to adapt to different contexts, foresee users' intentions, provide recommendations accordingly and protect users' privacy at the same time. In this paper we present our research on context-awareness applications, context-recognition techniques and privacy and trust. To demonstrate the feasibility and the progress in this area, we have developed a set of representative applications, two of which are presented in this paper. This perspective is the first glimpse of the forthcoming applications and services in the mobile domain. In the discussion, we summarize the many possibilities and describe the challenges for the research brought in by these new topics.

I. INTRODUCTION

Analyzing and building novel mobile applications and services were the goals of the European project MobiLife [11]. The focus of our work was to create applications and services able to adapt and accommodate users' needs according to their context and preferences. Context-awareness is believed to be one of the most promising forthcoming enhancements of mobile technology enabling both personalization and ubiquity of mobile services: it gives the user what he wants, anywhere he needs it, and in the best possible way. Mobile devices, and in particular mobile phones represent an attractive platform to develop and experiment new concepts and applications that can be gradually introduced into users' everyday lives. These devices are pervasive, they are with us most of the time, during the day and night and are always on. The possibility to create applications that exploit these characteristics on top of the devices software platform opens the way for *next-generation mobile applications*.

A new range of applications and services are possible by including context in the applications intelligence and reasoning. However, the underlying technologies and more importantly their integration make this simple vision a complex problem. In this paper, we present research topics that attempt to break through this complexity and we show some concrete applications.

The outline is as follows. We first give an overview of the general software architecture, its general principles and main drivers in section II.. We then describe the area of research we have targeted, namely, we will present methods used to handle context-awareness, section III., and privacy and trust, section IV.. Two selected applications called the *Family Maps* and the *Context Watcher* are described in section V.. The discussion, in section VI., presents the results achieved, summarizes the work done and points to the future work and challenges we believe are important.

II. MOBILIFE ARCHITECTURE OVERVIEW

The applications presented in this paper are based on the architecture developed during the MobiLife project [11, 3], which was designed to be user-centric and exploit the particularity of mobile devices. Context awareness, personalization (including personalized multi-modal interfaces), and privacy and trust were the key requirements.

With information about user context and preferences the software infrastructure allowed for the creation of intelligent applications which exploit, reason, and infer our needs in different circumstances. An important example is the context-awareness component and its distributed *Context Providers*. The applications used *Context Providers* to communicate their context or query about services provided by them (e.g. temperature in Helsinki). The architecture allowed to easily add new context sources and context providers (external services) to increase the application capabilities.

In the MobiLife project, applications targeted mobile devices, Nokia mobile phones as well as HP PDAs. Web Services (developed with Java, .NET, PHP) were used on the network side to store the applications' data, and deliver enriched information and functionalities. Applications were created in Python for Symbian S60, Java or C#, and had to comply with the API provided by the network services as described by the WDSL language [15]. Alternatively, HTTP requests with POST and GET methods, using the REST architectural style [7] were used. The latter option is even more attractive in devices with limited capabilities. Additionally, we experimented the use of external wireless Bluetooth sensors (e.g. GPS, heart monitor, etc.).

III. CONTEXT-AWARENESS

One of the goals of mobile or ubiquitous computing is to enable devices to sense changes in their environment and automatically adapt to these changes based on user needs and preferences. We refer to this ability as context awareness. Consider a context aware device: ideally, it should sense the users' context and react in an appropriate manner that facilitates the user inter-

1-4244-1144-0/07/$25.00©2007 IEEE

action with the device. A context-aware mobile device would help to offset the limitations of the constrained user-interface (i.e. limited keypad and screen) of mobile devices. In order to understand context and develop context-aware applications in mobile devices, the first step is to define what we mean by a users' context. Dey's following definition of context [4] is often quoted in the literature:

[Context is] 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 applications themselves.

This definition is a formal statement of our intuitive understanding of context. In situations where very few sources of context are identified, and when these pieces of context are obvious to us, it is relatively easy to create a system or an application that reacts to context. Mobile context-aware applications frequently rely on the use of location information as a primary source of context [2, 13], including the Context Watcher application which is described in section B.. Location information (from GPS, GSM cell id, short-range radio beacons) is often at the same time very powerful, easy to understand and can be used without additional signal processing. But what happens then when more sources of context data are available? For example a mobile device could provide the list of surrounding Bluetooth devices, temperature, pressure, accelerometer information, the users' heart beat, or data coming from sensors placed in the environment. At that point, it becomes much more complicated to intuitively understand how this context can or should influence the application, and to design a generic solution in which several context sources are used to describe the context. During the course of the project we have formalized a practical definition of context that is both generic and flexible. Using a statistical approach we proposed to define contexts as clusters in the data [1]. Figure 1 shows a visualization of the clusters in a set of sensor data recorded while a user was going from work to home while first walking to the bus station, then sitting in the bus, then being at home. The dataset contains records of the location, temperature, humidity, noise level, atmospheric pressure and user activity. The dark blue areas represent data cluster centres, and the lighter blue and yellow orange colors represent separations between clusters. Plotted on the visualization are data samples from the dataset sensed in different user contexts. It is clear in figure 1 that data samples from a specific user context are clustered together, and a cluster contains only data samples from a specific user context. Using this approach, it is possible to recognize clusters in the data that are unambiguously associated to a high-level context or situation.

Figure 1: Visualization of the data clusters in a sensor dataset recorded by a single user. The . show data samples when the user is walking to a bus stop, the $*$ show data samples when the user is sitting in a crowded bus and the *?* show data samples when the user is at home.

IV. Privacy and trust

Next-generation applications for mobile devices are typically user-centric and personal. They process and exchange personal user data or information about the user's context. Therefore it is important that adequate mechanisms are in place to protect the user's privacy and avoid the misuse of the system for invading the user's privacy sphere.

A user-centric privacy and trust framework for controlling the access to user data is required, which still enables flexible information exchange among users [14]. In such a framework, the basic principle should be that users are in full control of how their data items are shared with other users and applications. Access to user data can be controlled with the help of privacy policies defined by the user. The policies mandate which data may be shared with whom and under which circumstances.

In the MobiLife project the policy control has been implemented with so-called *Trust Engines*. Trust Engines are secured components that control access to all data items held in data storages. Since user data may be distributed to several different data storages, the access control function must also be distributed. Therefore each data storage has a dedicated Trust Engine acting as a policy enforcement point, which evaluates each data access request and determines whether the request can be granted or not. The users can use their personal User Trust Engines to define, modify and review their personal privacy policies. The policies are then replicated to such Trust Engines which hold custody of data items belonging to the user. This kind of access control architecture was adopted in order to deal with the fact that users are not always on-line and therefore the system cannot rely on the User Trust Engine to be able to act as an enforcement point at all times.

Although the Trust Engine approach is a workable solution from a technical point of view, it is clear that some usability related challenges remain with this approach. Even in simple scenarios users have to deal with a dozens of other users and applications, which can be grouped and classified in numerous different ways. Each of these security associations between users and applications would have to be dealt with by using dedicated privacy policies, resulting in hundreds if not thousands of user-specific privacy policies. It is quite obvious that specifying and managing such a large number of privacy policies manually by the user is not a realistic option. Especially the visualization of a large number of privacy policies on

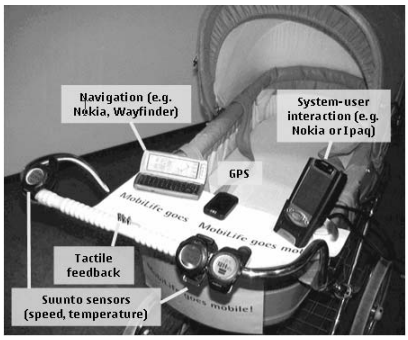

Figure 2: The baby pram used in the user evaluation phase of Family Maps

Figure 4: Posting context annotated pictures in Flickr

a hand-held device is challenging. A topic for further research therefore remains regarding how to provide the user with direct control over her privacy preferences without overwhelming the user with excessive privacy policy details.

V. Applications

A. Family Maps

Family Maps targets the needs of families with young children and assist them by providing special content and notifications. The interactive system supports communities and makes possible to leave notes for other parents. As an example of possible use case, a mother browses Family Maps at home and notices a new special offer for diapers in the local shop. When she is in proximity of the shop, she receives a notification of the same special offer. In the situation where she would like to post a note about another offer, she can create a reminder about the place in the Family Maps and write a note once she gets back home.

Figure 2 shows the baby pram that was used for the user evaluation. The clutter of various devices in the pram illustrates one of the main practical challenges in context-awareness and multi-modality research as prototyping and implementing the basic and often simple features of an application may require a complex setup, as no device fulfills all technical needs in terms of sensing, input or output modalities, connectivity and software and programming capabilities.

B. Context Watcher

The Context Watcher [10] is an application written in Python for the Nokia S60 platform, that allows users to record and label their location, upload it to a server and share it with their buddies. The location information can also be used as a meta data tag for photos. The Context Watcher has also access to other services delivered by various Context Providers such as local weather information, maps, collocation of buddies. GSM and 3G networks cells are uniquely identified by the Mobile Country Code (MCC), Mobile Network Code (MNC), Location Area Code (LAC) and Cell Identification (Cell id). By combining cell ids with GPS information, it is possible to geospot network cells, that is, to associate a GPS coordinate to each cell. The Context Watcher, see figure 3, has access to the GSM/3G network location information and may have access to the GPS longitude and latitude coordinates in the case where a Bluetooth GPS receiver is connected to the device. This location information is sent to the application Web Service, along with other information such as the user identification, time, battery level, user-input data such as mood, or current activity.

The server manages all the information and fulfills several purposes: it updates the geospotting database, runs a clustering algorithm to aggregate location records of users into meaningful places (e.g. home, office, commuting), collocates different users when in the same place; and delivers to the Context Watcher the current location of the users' buddies, provides maps and geocoding data. Photos taken in the Context Watcher can be published to the Flickr service [9]. The server automatically annotates the title, description and tags based on relevant context (Figure 4). The server can also automatically update the device users' blog. A process runs at the end of every day and sends a post to the users' blog with the photos of the day, and statistics on where the user was, and whom he met.

VI. Discussion

Mobile devices, in particular mobile phones, are successfully used to create intelligent applications that are able to understand user needs, context and have the ability to adapt and provide recommendations [12]. With mobile phones we have experimented with the use of location, time, used additional sensors and external web services. Sensors included the use of accelerometers, temperature sensors, heart monitors. In summary, the applications shown in this paper are examples of applications which can be created now with today's technology. The use and process of many different sources of information makes it possible to predict and infer users' needs to create applications that adapt to different circumstances and context [8].

However additional expertise in context adaptation and user profiling needs a more in-depth investigation. To make sense of the vast amount and variety of information collected we have to have better algorithms and techniques for better reasoning and processing. Especially in mobile phones, we need to find solutions that take into account the limited processing power, battery-life and capabilities of these devices. A distributed architecture with external processing and data sources must be considered. However, the success of these services depends greatly on how well all the necessary functional components

Figure 3: Context Watcher - selected views of the phone application

are integrated together. The platform created in MobiLife provided a design to support distributed data sources and data processing elements.

Another challenge we found concerns the fragile boundary between the need for users' data and the requirement to preserve users' privacy. Even though the potential of context-aware applications are immense and will have a beneficial impact on our lives, their impact on our privacy must be evaluated. Reality mining is already happening and it is not an hypothetical threat anymore [5, 6]. Our social networks, habits and preferences can be revealed and be used for scopes that were not initially intended for. In this context, research on privacy must consider the implications of these new applications and devise ways to protect users' privacy while at the same time allow the full potential of these applications. To provide the right balance is the main challenge. The Trust Engine concept introduced in the MobiLife represents an example through which users will have the ability to fully control how information about them is released to external parties. A lot of further research is, however, still required. For example hiding the inherent complexity required for setting up and managing the privacy policies of a user in any real-life scenarios is still a question that remains to be solved in order to make the privacy protection framework usable in day-to-day life.

VII. Conclusions

In this paper we have presented some of the possibilities and challenges of *next generation applications* in the mobile domain. These user-centric applications used various data sources to recognize users' context and provide the best user experience. While processing the use of personal information, users' privacy had to be preserved and this was one of the big challenges. From the research point of view, we have presented a statistical approach to context recognition. We included two examples of context-aware applications, the Context Watcher and the Family Maps.

In our job, the practical approach to research has had several benefits. The development of prototypes and the use of context data have offered a great opportunity to understand, in practice, the evolution of next generation mobile applications. Future research will concentrate on extending and using these applications to provide an understanding on context-awareness and privacy and trust.

References

[1] A. Battestini and J. Flanagan. Analysis and cluster based modelling and recognition of context in a mobile environment. *proceedings of MRC 2005*.

[2] J. Burrell, K. Gay, Geri, and N. Kubo, Kiyoand Farina. Context-aware computing: A test case. *Proceedings of the International Conference on Ubiquitous Computing*, pages 1–15, 2002.

[3] C. Del Rosso, M. Klemettinen, B. Mrohs, S. Steglich, and C. Rack. The mobilife service infrastructure. *Proc. of the 15th IST Mobile Summit*, June 2006.

[4] A. Dey. Understanding and using context. *Personal and Ubiquitous Computing*, 5(1):4–7, 2001.

[5] N. Eagle. Machine perception and learning of complex social systems. *Ph.D. Thesis, Program in Media Arts and Sciences, Massachusetts Institute of Technology*, June 2005.

[6] N. Eagle, A. Pentland, and D. Lazer. Inferring social network structure using mobile phone data. *Proceedings of the National Academy of Sciences of the United States of America, (PNAS) (In submission)*, 2007.

[7] R. T. Fielding. Architectural styles and the design of network-based software architectures. *PhD thesis, UC Irvine*, 2000.

[8] J. A. Flanagan. Unsupervised clustering of context data and learning user requirements for a mobile device. In *CONTEXT*, pages 155–168, 2005.

[9] Flicker. http://www.flickr.com.

[10] J. Koolwaaij, A. Tarlano, M. Luther, B. Mrohs, A. Battestini, and R. Vaidya. Contextwatcher sharing context information in everyday life. *IASTED International Conference on Web Technologies, Applications, and Services (WTAS2006)*, July 2006.

[11] MobiLife. http://www.ist-mobilife.org.

[12] S. Moloney. Simulation of a distributed recommendation system for pervasive networks. In *SAC*, pages 1577–1581, 2005.

[13] Ubicomp 2006. The eight international conference on ubiquitous computing. 2006.

[14] R. Van Eijk, O. Coutand, and S. Holtmanns. Sharing of preferences and context in groups of mobile users. *Workshop on Mobile Social Software, (CHI2006)*, April 2006.

[15] W3C. Web services description language (wsdl). *http://www.w3.org/TR/wsdl*.