Challenges in the Development of Context-Inference Systems for Mobile Applications

André C. Santos[*]
Technical University of Lisbon
Lisbon, Portugal
acoelhosantos@ist.utl.pt

João M. P. Cardoso
Faculty of Engineering,
University of Porto
Porto, Portugal
jmpc@acm.org

Pedro C. Diniz
Technical University of Lisbon
Lisbon, Portugal
pedro.diniz@ist.utl.pt

Diogo R. Ferreira
Technical University of Lisbon
Lisbon, Portugal
diogo.ferreira@ist.utl.pt

ABSTRACT

Communication and service providers strive to offer (and charge for) added-value services that leverage contextual user information such as current location and/or activity (e.g., `walking`, `running`, `sitting`). Obtaining this information and accurately determining the user context poses many practical and implementation challenges, making the development of context-aware applications notoriously difficult. In this paper we present our perspectives and experiences on some of the challenges encountered when developing a context-inference system based on mobile phones, both from a software and hardware perspectives. Specifically, we focus on issues related to handling contextual information, using sensor-based hardware prototypes and developing context-inference software for mobile applications.

Keywords

Context, context-awareness, context-inference, mobile devices, programming challenges.

1. INTRODUCTION

Context-awareness is a central issue in ubiquitous and wearable computing [9], and currently a very active research topic. The ability to understand the contextual situation of users, devices and surrounding environment, enables the use of this information in computational systems to provide overall better results in their designated tasks (e.g., GPS navigation using traffic data; music selection according to the user's mood; efficient automatic watering gardening systems using temperature and humidity data of the soil).

Context-awareness manages information entities known as contexts, which have been described in the literature in various, often similar ways. For instance, Abowd *et al.* [1] defined context as any information that can be used to characterize the situation of an entity; be it a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and the applications themselves. Context-inference is thus the process of detecting or inferring the context from input data sources (data providers or sensors), and is the primary feature for achieving context-awareness. This process can be fairly sophisticated, incorporating methods and techniques that range from simple statistical operators to complex machine learning algorithms.

Recently we have observed an increasing number of applications and services (e.g., location-based services) that use the contextual information to enhance their value, productivity and aesthetics. Trends suggest that this contextual intelligence will be incorporated in everyday objects like clothes, vehicles, picture frames, etc. Despite the various research efforts in the area of context awareness (e.g., [3, 4, 6, 8, 15]), the development of systems that gather and use context information still presents numerous challenges. These challenges are naturally exacerbated when these context-aware systems are intended to be executed on mobile devices, using other, often distributed, embedded systems as data sources for context inference.

In this paper we describe some of the challenges we encountered during the development and evaluation of a user-activity context-inference system [16], both from a hardware and a software point-of-view. The system consists of a vest prototype equipped with several of sensors that transmit data wirelessly, via bluetooth, to a mobile phone, which is responsible for its preprocessing and context-inference. By describing the challenges we faced and sharing insights from our own experience, we hope to provide the community with an account of the most common issues in the development of mobile context-inference systems.

The remainder of the paper is organized as follows. Section 2 describes the prototype system we have developed. Section 3 enumerates and describes common challenges when

[*]This work was partially funded by the Portuguese *Fundação para a Ciência e a Tecnologia* (FCT) under grant SFRH/BD/47409/2008.

developing context-inference systems. We then conclude and outline possible solutions and/or research directions that address various of the challenges discussed.

2. UPCASE PROTOTYPE

Our context-inference system [16] consists of a wearable hardware prototype and a software processing unit deployed on a mobile phone. The main goal of our system is to create a light-weight and thus portable mobile context-inference prototype that allows the accurate identification of users most common physical activities (e.g., `walking`, `running`, `working`, `driving`) using typical sensors (e.g., accelerometers, GPS, sound, time). The software processing unit relies solely on the mobile phone and includes the capability of publishing the user's context information to interested applications or services.

Sensors included range from physical to virtual. Physical sensors such as sound, temperature, accelerometers, humidity and light sensors, as well as virtual sensors (e.g., time of day) are used. Physical sensors are aggregated on a BlueSentry[1] sensor node, contrasting with the virtual and logic sensors which exist within the mobile phone. Software development was accomplished using Java ME[2] (J2ME). Figure 1 presents the prototype embedded on a vest garment and being used to infer user activities such as e.g., `walking outside in a park during the day`.

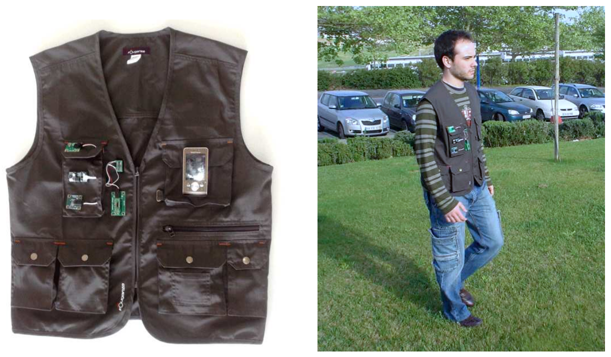

Figure 1: UPCASE context-inference system for the identification of user activities.

The system is fully operational and currently serves as a testbed for several context-aware studies, including the application to social networking [16].

3. DEVELOPMENT CHALLENGES

We can identify the challenges faced in the development of mobile context-inference systems and their corresponding applications in the following areas:

- Context information issues: management of context hierarchy, storage and sharing models, information quality and context inference confidence/accuracy.
- Hardware issues: Requirements, sensors to be used, overall design and placement/organization.
- Software issues: Choice of platform, separation between application logic and context-inference code, execution of complex and demanding algorithms on constrained mobile devices and problems with debugging.
- Cross-cutting and system-level issues: Energy, communication, scalability and privacy.

[1]http://www.rovingnetworks.com/bluesentry.php
[2]http://java.sun.com/javame/

These challenges cover mobile devices issues and requirements, context information use and quality, as well as prototype sensor data problems. In the following sections we describe these challenges in further detail.

3.1 Context Information Issues

Context information can aggregate data across several stages of processing. Furthermore, context information can be organized in several ways, such as to provide a hierarchy (e.g., location context `at home` associated with activity context `watching television` produces a higher level context `watching television at home`), for example. Context information needs to be modeled to accommodate its characteristics, data and relations. It is very important to define what the context information is, how it changes and later how one can use it in real-world applications. Further importance of context understanding was reported by Pascoe *et al.* [13], who emphasized the usefulness of context-awareness; the commonly incorrect perception of context and how rich and complex it can be; and the fact that context is an attribute of entities so the more contextual information, the more entity characterization is possible.

Several context models have been described in the literature to handle context information. An efficient model for handling, sharing and storing context information is essential for a working context-aware system. In addition, a context model is needed to store contextual information in a machine processable form [3]. The most relevant context modeling techniques are key-value models, markup scheme models, object-oriented models, logic-based models and ontology-based models [17]. No context model can be assumed to be a standard, and a deep understanding of the context problem is essential for choosing the right model [5].

Within a model, a context needs to be correctly described by means of a set of attributes [3]. Commonly used attributes include: *context type* which is the category of context such as temperature or time; *context value* which consists on the raw data gathered by a sensor and the measurement unit; *timestamp* which marks the context information with date and time that describe when the sensing took place; *source* containing information on how the data was gathered, namely, from which sensor it came; and *confidence* which describes the (un)certainty of the recognized context.

Within the context-inference process, the inferred context needs to exhibit quality and confidence. If this information is incoherent or incorrect, system functionality is lost (e.g., `"brushing teeth on the highway"` is an unlikely context to occur). Uncertainty comes from several sources but is largely due to the lack of precision and accuracy of sensors interpreting the environment as well as due to errors and probability characteristics of the inference techniques used. Dealing with the uncertainty of the inferred context is an important and complex task. As user context-aware systems are inherently uncertain, approaches to tackle this issue and improve system reliability are required (e.g., [2, 7, 18]).

Also, context identification needs to be aligned with the system and platform used. Real-time requirements and specific context requirements for identification critically influence the choice of methods used for context inference.

3.2 Hardware Issues

To infer context information, data sources or sensors have to be used, ranging from physical sensors (e.g., temperature sensor), to virtual sensors (e.g., time of day) or even logical sensors (e.g., traffic conditions) which are all specialized data providers for determined properties. A hardware prototype – equipped with an array of sensors, processing power for real-time, storage capabilities, wireless communication, and a minimum battery life for one day – are reasonable requirements for context-inference systems. However, the combination of all these requirements brings several challenges during system development.

The design of a hardware prototype, meant to be portable, has to be comfortable and unobtrusive to be easily worn. However, sensors, sensor nodes, sensor connections and batteries have all to be considered so that the hardware prototype is not too heavy, too big and too cluttered. Device placement is very important: wrist, waist or backpack are simply some examples of the wide range of available options. Ultimately, it all depends on the type of context, object of identification. For example, considering a soccer player: using a limited number of sensors, it is more useful to place sensors on the player's legs than on the player's arms; since information on soccer is mostly acquired from leg movement. In addition, when designing pervasive environments, concealment of the hardware in the garments is very important, often obligating accommodating all sensors on a single cluster.

The use of a vest-garment as our latest prototype equipment allowed an easy and fast user setup. However, several connections from and to the sensors had to be made, requiring that sensor connections be carefully placed in order to avoid disconnections that may occur during user activity. Connection problems translated into erroneous and missing sensor acquisitions.

When using multiple off-the-shelf sensors from different manufacturers, we realized that specifications varied (mostly their sampling frequency) and we had to consider the difference in frequency on our software components. Individual sensor calibration and fine-tuning were accomplished to understand the output and the behavior of the different sensors. Keeping the hardware components from the same manufacturer can reduce this extra auxiliary work.

3.3 Software Issues

The software for context inference includes the engine that interacts with the hardware devices (*i.e.*, sensors and actuators) and provides the methods for context recognition. Various challenges exist if the software component is to be executed on computationally constrained devices such as mobile phones. Despite their increase capabilities, mobile phones are still more limited than traditional desktop or portable computers. Memory and processing power are reduced, and special characteristics such as resolution of the screen, I/O operations, physical size, etc., have to be taken into account. Also, developing a generic software solution that operates with a wide range of devices and platforms is very difficult (e.g., Android, iPhone OS, Symbian C++ and J2ME).

One great concern is the separation of application code from the context-inference engine. Most context-aware applications embed the context interpretation logic inside of the applications themselves, making the development and future maintenance extremely difficult. Also, the development of context-inference systems involves a large amount of boilerplate code even for simple tasks, unless a specialized architecture, framework or middleware is used (e.g., Context Toolkit [15], CASS [8], JCAF [4]). In addition, the overhead of sensor data, processing and context identification rules, and triggering actions, needs to be addressed. Regarding sensor data, the software needs to cope with sensor problems, such as malfunctions and missing data values.

Many context-inference algorithms have been developed or adapted from other research areas. Adaptation cannot in general be directly applied to mobile devices, due to the their limited characteristics. Adaptation and customization/optimization of promising existing algorithms often requires a non-trivial amount of effort. As with other research prototypes (e.g., [12]), our system uses a custom adaptation of an existing algorithm based on decision trees as its context-inference engine.

Portability typically requires the use of wireless communication, which in our system consisted of bluetooth communication. Problems with bluetooth, namely with data transmission, losing connections and interferences, were observed, which were address in software.

Debugging is a difficult and laborious task. The dependency on multiple sensors requires debugging to be done on the device itself. The standard approach for debugging is complex and very slow, requiring many code compilations, deployments and executions. When possible, emulators with profilers help with corrections and optimizations, by providing a better perception of execution time and memory use.

Lastly, if the software application requires user interaction, context awareness must be perceived by the user as neither intrusive nor useless. In this respect, human-computer interaction has to be reduced to its simplest form, in order not to overwhelm the user with complex system configurations, instructions and feedbacks.

3.4 Cross-Cutting and System Issues

As a whole, there are challenges that affect all the components of a portable context-inference system. First, managing energy consumption is a major concern that affects both the system hardware and software. Hardware components must be chosen understanding their power requirements so that batteries have to be well-dimensioned to provide the best size/power relation. In software, energy-aware programming is also essential, as for example more complex algorithms in general lead to larger energy consumption.

Scalability is a very important issue in the context of a high number of users and activities. Different users, can use the system distinctively, perform similar activities in different manners or perform completely novel activities. The system also needs to support a considerable amount of different activities. This is difficult to accomplish as different user activities can be very similar and are therefore hard to be distinguished with high accuracy.

Communication issues are cross-cutting as they take an important role in all the mentioned components. Different means of communication (e.g., Wi-Fi, Bluetooth) have considerations on hardware components, software interfaces and energy consumption. Their frequency of use, sleep and operation time are the main issues to take into account.

Lastly, since context information can contain user private data (e.g., user emotions, current activity, health details, social networks), context-inference systems need to enforce privacy policies to ensure the security of this information. Without privacy guarantees, users will be reluctant to use context-aware systems. Several efforts to ensure privacy on contextual information have been developed for context-aware systems, namely [10, 11, 14].

4. CONCLUSIONS

In this paper we have outlined and described common challenges that developers face when implementing context inference systems for mobile applications. The challenges presented are not exhaustive and the corresponding issues are deeper and broader than what could be described here. We focused on what were the most important and most time-consuming challenges we were faced when developing our context-inference system prototype. Overall these challenges should be met by technological frameworks able to integrate the multi-disciplinary facets of context-aware systems and applications. In our view the challenges presented can be tackled within the scope of the following solutions:

- Domain specific programming languages to assist development of these systems (e.g., dealing with tasks, synchronization, sampling, preprocessing).
- Choice of algorithms (e.g., machine learning classifiers) based not only on their performance but also on their computational complexity.
- Middleware and virtual computing engines allowing an easy way to plug-in software components for data acquisition and preprocessing.
- Standards and flexible schemes to describe the interaction with hardware components.
- Simulation environments for multiple devices (e.g., emulators) and ability to mimic real scenarios.

We hope that the challenges described in this paper which we experienced in the development of our context-aware mobile applications, will help other developers in their projects, preparing them with a broader knowledge of the issues and a clearer perception of their difficulties, thus hopefully increasing development productivity.

5. REFERENCES

[1] G. Abowd, A. Dey, P. Brown, N. Davies, M. Smith, and P. Steggles. Towards a better understanding of context and context-awareness. In *Proc. of the 1st Intl. Symp. on Handheld and Ubiquitous Computing (HUC'99)*, pages 304–307, London, UK, 1999. Springer-Verlag.

[2] S. Antifakos, N. Kern, B. Schiele, and A. Schwaninger. Towards improving trust in context-aware systems by displaying system confidence. In *MobileHCI '05: Proceedings of the 7th international conference on Human computer interaction with mobile devices & services*, pages 9–14. ACM, 2005.

[3] M. Baldauf, S. Dustdar, and F. Rosenberg. A survey on context-aware systems. *International Journal of Ad Hoc and Ubiquitous Computing*, 2(4):263–277, 2007.

[4] J. E. Bardram. The java context awareness framework (jcaf) - a service infrastructure and programming framework for context-aware applications. In *Pervasive Computing*, pages 98–115, 2005.

[5] C. Bolchini, C. A. Curino, E. Quintarelli, F. A. Schreiber, and L. Tanca. A data-oriented survey of context models. *SIGMOD Rec.*, 36(4):19–26, 2007.

[6] G. Chen and D. Kotz. A survey of context-aware mobile computing research. Technical report, Dartmouth College, Hanover, NH, USA, 2000.

[7] A. K. Clear, S. Dobson, and P. Nixo. An approach to dealing with uncertainty in context-aware pervasive systems. In *Proc. of the UK/IE IEEE SMC Cybernetic Systems Conference*, 2007.

[8] P. Fahy and S. Clarke. Cass – a middleware for mobile context-aware applications. In *Workshop on Context Awareness, MobiSys*, 2004.

[9] T. Huynh and B. Schiele. Analyzing features for activity recognition. In *Proc. of the 2005 joint conference on Smart objects and ambient intelligence (sOc-EUSAI '05)*, pages 159–163, New York, NY, USA, 2005. ACM.

[10] X. Jiang and J. A. Landay. Modeling privacy control in context-aware systems. *IEEE Pervasive Computing*, 1(3):59–63, 2002.

[11] J. Loyall, P. Pal, K. Rohloff, and M. Gillen. Issues in context-aware and adaptive middleware for wireless, mobile networked systems. BBN Technologies, 2009.

[12] J. Pärkka, M. Ermes, P. Korpipää, J. Mäntyjärvi, J. Peltola, and I. Korhonen. Activity classification using realistic data from wearable sensors. *IEEE Tran. on Information Technology in Biomedicine*, 10(1), January 2006.

[13] J. Pascoe, N. Ryan, and D. Morse. Issues in developing context-aware computing. In *Proc. of the 1st Intl. Symp. on Handheld and Ubiquitous Computing (HUC'99)*, pages 208–221. Springer-Verlag, 1999.

[14] D. Riboni, L. Pareschi, and C. Bettini. Privacy in georeferenced context-aware services: A survey. In *Privacy in Location-Based Applications*, pages 151–172, 2009.

[15] D. Salber, A. Dey, and G. Abowd. The context toolkit: Aiding the development of context-enabled applications. In *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems (CHI'99)*, pages 434–441, New York, NY, USA, 1999. ACM.

[16] A. C. Santos, J. M. P. Cardoso, D. R. Ferreira, P. C. Diniz, and P. Chainho. Providing user context for mobile and social networking applications. *Elsevier Pervasive and Mobile Computing*, 2010.

[17] T. Strang and C. Linnhoff-Popien. A context modeling survey. In *Workshop on Advanced Context Modelling, Reasoning and Management, UbiComp'04 - The 6th Int. Conf. on Ubiquitous Computing*, 2004.

[18] B. Truong, Y.-K. Lee, and S.-Y. Lee. Modeling and reasoning about uncertainty in context-aware systems. In *Proc. of the IEEE Int. Conf. on e-Business Engineering (ICEBE'05)*, pages 102–109. IEEE, 2005.