ECAI 2008
M. Ghallab et al. (Eds.)
IOS Press, 2008
© 2008 The authors and IOS Press. All rights reserved.
doi:10.3233/978-1-58603-891-5-668

AI on the Move: Exploiting AI Techniques for Context Inference on Mobile Devices

Adolfo Bulfoni and **Paolo Coppola** and **Vincenzo Della Mea** and **Luca Di Gaspero** and **Danny Mischis** and **Stefano Mizzaro** and **Ivan Scagnetto** and **Luca Vassena** [1]

Abstract. Context aware computing is a computational paradigm that has faced a rapid growth in the last few years, especially in the field of mobile devices. One of the promises of context-awareness in this field is the possibility of automatically adapting the functioning mode of mobile devices to the environment and the current situation the user is in, with the aim of improving both their efficiency (using the scarce resources in a more efficient way) and effectiveness (providing better services to the user). We propose a novel approach for providing a basic infrastructure for context-aware applications on mobile devices, in which AI techniques (namely a principled combination of rule-based systems, Bayesian networks, and ontologies) are applied to context inference. The aim is to devise a general inferential framework to easier the development of context-aware applications by integrating the information coming from physical and logical sensors (e.g., position, agenda) and reasoning about this information in order to infer new and more abstract contexts. In previous context-aware applications, most researches focused almost exclusively on time and/or location and other few data, while the same contexts inference was limited to preconceived values. Our approach differs from previous works since we do not focus on particular contextual values, but rather we have developed an architecture where managed contexts can be easily replaced by new contexts, depending on the different needs. Moreover, the inferential infrastructure we designed is able to work in a more general way and can be easily adapted to different models of applications distribution. We show some concrete examples of applications built upon the inferential infrastructure and we discuss its strengths and limitations.

1 INTRODUCTION

Recently we have assisted to the widespread of mobile devices such as PDAs, smartphones, etc. Due to a very rapid evolution trend, these devices have become more and more similar to traditional computers, both in terms of capabilities and computational resources. However, differently from traditional computers, these devices are usually employed "out there" in the real world, e.g., while the user is busy doing other activities (such as walking down a street, shopping, and so on). The scientific community is looking for new approaches to application development and user-device interaction management.

A key-role in these new approaches is played by the notion of *context*, that is roughly described as the situation the user is in. This concept encloses important information that could be used to affect the capabilities of mobile devices, adapting them to the user's needs. Context awareness allows individuals to interact with systems that are aware of the environmental state (e.g., location, workgroup, activity) and computational state (e.g., applications, devices, services) of an individual [14].

One of the most crucial aspects in context-aware applications is the inference of the user's context. In this paper we propose a new approach based on the combination of three classical AI techniques, namely rule-based systems, Bayesian networks, and ontologies. The feasibility of this approach is then demonstrated by means of example applications developed using the MoBe architecture [7], a framework for context-aware applications on mobile devices.

This paper is structured as follows. After a brief introduction to context-aware computing and the presentation of the framework we intend to exploit for our experiments (Section 2.1), we give a detailed description of the proposed approach to context inference (Section 3). In Section 4 we present a concrete implementation that demonstrates the feasibility of our idea. Finally we present some conclusions and future work.

2 RELATED WORK

2.1 Context-aware computing

Context-aware computing can be defined as the use of context in software applications, where the applications adapt to discovered contexts by changing their behavior [6]. The concept of context is still a matter of discussion and through the years several different definitions have been proposed. They can be divided in *intensional* definitions and *extensional*.

Extensional definitions present the context through a list of possible associated values. In the first work that introduces the expression *context-aware* [12], the context is represented by the location of the user and the surrounding objects. In a similar way, Brown *et al.* [5] define context as location, proximity to other people, temperature, day and hour, etc.

Intensional definitions present the concept of context more formally. In [1] the context is defined as *"any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves"*. For Brazire and Brezillion [3], *"the context acts like a set of constraints that influence the behavior of a system (a user or a computer) embedded in a given task"*. This definition moves from the analysis of a collection of 150 context definitions from several field of application like sociology, computer science, etc.

Extensional definitions seem to be useful in practical applications, where the abstract concept of context has to be made concrete. How-

[1] University of Udine, Italy, email: bulfoni@dimi.uniud.it, coppola@uniud.it, dellamea@dimi.uniud.it, l.digaspero@uniud.it, mischis@dimi.uniud.it, mizzaro@dimi.uniud.it, scagnett@dimi.uniud.it, vassena@dimi.uniud.it

ever, from a theoretical point of view they are not properly correct, as the context cannot be outlined just by some of its aspects. On the other hand intensional definitions are of little use in the practice, despite they are theoretically satisfying.

Moving from these considerations, we can understand why in previous context-aware applications the notion of context mainly included just a little number of information. Most researches, for example, focused almost exclusively on time and/or location and other few data [14]. More complex approaches tend to combine several contextual values to generate new contextual information. In [1] *primary* contexts as location, entity, activity, and time, act as indices into other sources of contextual information. Similarly, in the TEA Project [13] Schmidt *et al.* use a resolution layer to determine a user's activity starting from basic contextual information.

As in the previously cited works, we want to combine contexts to determine new, more abstract contexts. Differently from them, however, we do not focus only on particular contextual values, but we develop an inferential infrastructure able to work in a general way.

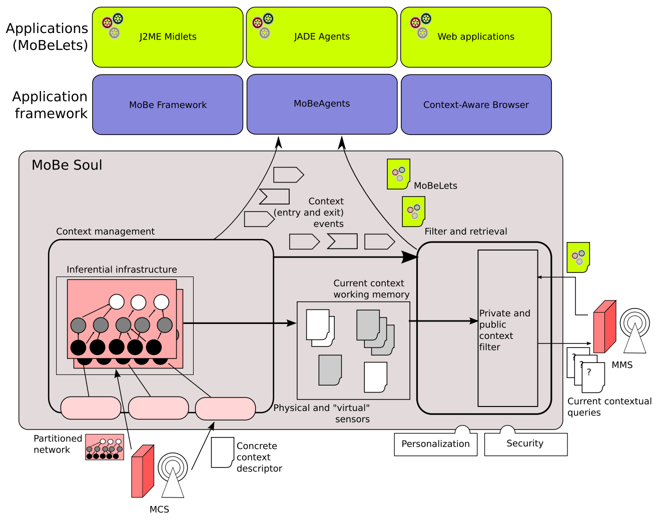

Figure 1. General architecture of MoBe platform.

2.2 The MoBe architecture

MoBe is a general architecture for context-aware distributed applications on mobile devices based on the dynamic and automatic download, configuration, execution, and unload of applications on the basis of the user's current context. Rather than having the applications rigidly installed on a mobile device, the user and device context is used to obtain and start useful applications and to discard the not anymore useful ones. This way, a device is not limited to a set of predetermined functionalities, but allows to adopt those which are probably more useful for the user at a given time.

For example, when a person enters his home, his device provides automatically the application to control the household appliances. This application can be discarded (or just stopped) when the person leaves home. The device can turn into a TV remote controller while the user is watching TV, or it can turn into a cooking book while the user is cooking, etc.

MoBe architecture is presented in Figure 1 and is composed by the following three layers (from bottom to top):

MoBeSoul: is the middleware whose basic responsibility are to sense the surrounding environment, to perform the context inferences and to manage the retrieval of applications.

Application framework: consists of the software infrastructure for building the concrete mobile applications. Since the MoBeSoul component is completely general and can be adapted to different implementations, we currently have developed and tested three implementations based on (i) MoBe framework, an ad-hoc J2ME middleware, (ii) MoBeAgents, the extensions of a Multi-Agent framework, (iii) a *Context-Aware* browser, a browser extension that allows the development of contextual web applications.

MoBeLets: a basic context-aware application built upon this architecture.

The applications, called MoBeLets reside on the MoBeLet Server and migrate, transparently to the user, on her mobile device. Each MoBeLet presents a descriptor that holds the most important information related to the application, in order to make the retrieval easier (e.g., the type of task carried out) and to decide whether it is suitable or not for the mobile device of the user (e.g., information about the minimal CPU/memory requirements or the kind of needed peripherals/communication media).

Some ancillary servers exchange information with mobile devices in that environment. In particular the MCS (MoBe Context Server) exchanges information about contexts (and inferential networks), while the MMS (MoBe MoBeLet Server) provides MoBeLets' related information and the actual MoBeLets.

The workflow managed in the MoBe architecture is the following:

1. the MoBeSoul acquires information related to the user and the surrounding environment, by means of sensors installed on the device or through a Context Server;
2. from this contextual information, the MoBeSoul infers the user's context (and its likelihood);
3. the user's context is sent to the MoBeLet Descriptor Search Engine that looks for the MoBeLets most suitable for the user's context and sends their descriptors to the MoBeSoul;
4. on the basis of user preferences, the descriptors can be filtered again;
5. the remaining descriptors are used to obtain the applications from MoBeLet server. The MoBeLets currently executed on the device are managed on the basis of the user's context: when a context is not valid anymore, for example, the associated application can be stopped or discarded.

3 CONTEXT INFERENCE SYSTEM

3.1 Inferring abstract contexts from concrete contexts

The user's current context is composed by an undefined number of contextual values. Each value is described by two elements: an unambiguous ID and a probability value. We divide contextual values into two categories:

Concrete contexts: represent the information obtained by a set of sensors. These contexts can be read from the surrounding environment through physical sensors (e.g., temperature sensor), or can be obtained by other software (e.g., calendar) through logical sensors. Some examples are: "temperature: 20°C", "12:30", "meeting at 14:30", and so on. Concrete contexts are returned by the sensors and represent the input of the inferential mechanism.

Abstract contexts: represent everything that can be inferred from concrete contexts like, for example, "user at home", "user is shopping", etc.

The problem we are facing is therefore the definition of an inferential system capable to derive the abstract contexts from the concrete ones. Concrete and abstract contexts are the inferential system input and output, respectively. From a theoretical point of view, this difference is faded since the contexts cannot be unambiguously assigned to one or the other category: the context "temperature 90°C" can be a concrete contexts as it is obtained from a sensor, or it can be inferred by other contexts (e.g., "user in sauna"). The aim of the inferential system it to combine concrete contexts to determine abstract contexts and to combine abstract context to obtain new, more abstract contexts.

3.2 Two approaches for the inferential system

To develop our inferential system two approaches seem intuitively adequate and have been taken into consideration: rule-based systems and Bayesian networks.

As it is well known, a rule-based system [10] is a general mechanism for the knowledge representation and management. Although rule-based systems are a relatively simple model, they can be naturally adapted to the context-aware field. The left and right side of a rule can represent two contextual values and the rule suggests a connection between them: e.g. *IF* ⟨ *I'm in the bathroom* ⟩ *and* ⟨ *there is an high humidity level* ⟩ *and* ⟨ *there is a continuous sound* ⟩ *THEN* ⟨ *I'm having a shower* ⟩. Let us remark that rule-based systems allow to use variables, a feature that simplifies knowledge management.

Indeed, rule-based systems have already been used in the context-aware field. Bacon and colleagues [2] describe a multimedia system based on user location, where contextual information is represented as facts in a rule-based system. Zhang [15] proposes a framework for allowing user to program his context-aware application: a user can visually create the rules that are then combined with sensors data to adapt the application to user context. In [8] a rule-based system is used to trigger the actions depending on the registered contexts.

Bayesian networks [11] represent a model for the execution of inferences based on probability, and they can be easily adapted to the context-aware field as well. Each node can represent a contextual value, the edges representing dependence relationships between different contexts, while the probability distributions indicate the certainty related to contexts.

Bayesian networks have been used in the context-aware field mainly as system to determine the uncertainty of contexts. In [4] a Bayesian network is used to measure the efficiency of contexts derivation from rough sensors data while in [9] Bayesian networks are used to classify contexts related to a user's everyday activities.

Even if the use of rule-based systems is reasonable in context-aware computing, it presents a remarkable limit: they manage certain knowledge, whereas almost everything related to contexts is characterized by uncertainty. Because of the uncertainty management, Bayesian networks are more suitable than rule-based systems for contextual inferences. On the other hand, a rule-based system allows the use of variables, that allow to limit the dimension of the same inferential mechanism. For example instead of managing all temperature values we can simplify them to the three abstract values *temperature high, low, and normal* and use rules generalized with variables to map the real sensed temperature value on the three abstract ones. In our opinion, both the approaches are important and needed for a complete and functional system.

3.3 The inferential infrastructure

We propose a two stage inferential mechanism, where both rules and Bayesian networks are used. We define the combination of rules and Bayesian network as the *inferential infrastructure*.

The input to the inferential infrastructure is represented by concrete contexts. Concrete contexts are processed by a rule-based system in order to simplify the information and map them on the starting node of the Bayesian network, that represents the second and main stage of the inferential infrastructure, where abstract contexts are transformed into concrete ones. The combination of concrete and inferred abstract contexts is the user's current context (Figure 2).

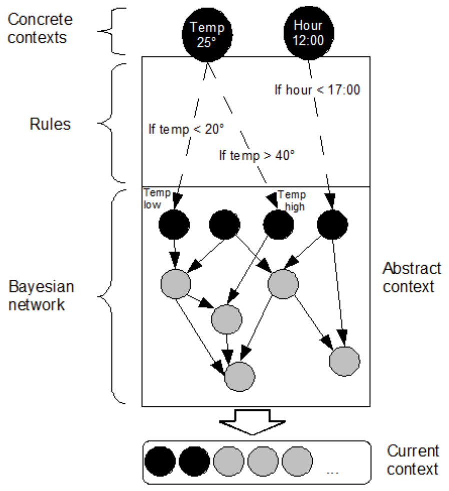

Figure 2. Inferential infrastructure model.

The following is a simple example. A concrete context is "sound 60dB". Through the first rule-based stage this information is mapped into the Bayesian network starting nodes. In this way we simplify the contextual information: instead of managing all the possible sound values we create an abstraction (i.e., "sound low", "sound high"). Without the rules (and the variables), if the management of all the single sound values was needed, we should have introduced in the Bayesian network a node for each sound value. Then the Bayesian network starts from this point to infer abstract contexts like "user is listening to music'.

3.4 Critical issues

We can identify four main limits of the basic inferential infrastructure we have presented so far.

The first limit concerns the size of the inferential network. Since we do not want to limit the inferences only on an a priori defined set of contextual dimensions, the network, in principle, should be omniscient and include information and inferential mechanisms for every possible situation. The possible contextual values that should be managed in such a case are innumerable and, as they have to be built into the inferential network, this would lead to a universal network, which would be unmanageable in practice.

The solution we adopt consists in splitting the inferential network into several subnetworks, on the basis of a single dimension of concrete context. For example, a hypothetical universal network can be split using location as an index: in this way we obtain several networks, each of them relevant to a particular location (e.g., one for home, office, car, etc.). These specialized networks have a smaller size, therefore they are more manageable. Furthermore, if they are still too complex they can be partitioned further.

Single inferential networks are acquired automatically on the basis of the value of the contextual dimension used for the partitioning. The mobile device will receive one or more networks from a remote server and will combine them to infer more precisely the user abstract context. For example, when the user enters her home, if the user location is selected as the partitioning dimension, the device will obtain the inferential network specialized on the contexts related to the home.

Location is just one of the dimensions that can be used to partition the inferential network; other dimensions could be time (e.g., a set of applications related to work is downloaded during working hours or according to an agenda) or a combination of location and time.

The second problem concerns the potential mismatch between concepts in the inferential networks and the concrete contexts (e.g., a network could execute inferences starting from a contextual value expressed as "temperature in Celsius degrees" while the temperature sensor on the device manages information in Fahrenheit degrees). These incoherences would of course lead to erroneous contexts inference. To avoid that, we define an ontology of concrete contexts, which provides a shared model for situations, sensors, provided values, and relationships among them. In this ontology, the sensors are categorized on the basis of the contextual information they provide. Similarly, in the development of the inferential network, its concrete nodes (or the starting ones) must also be categorized in the ontology. In this way the inferential engine is able to obtain the information needed by the inferential network through the appropriate sensors.

The third problem is a consequence of the previous two and it concerns the agreement on contexts representations. Indeed, it is possible to represent the same context in many different ways, leading to a troublesome matching between the context descriptions in the inferential network and those in the specific application descriptors. Again, a shared ontology allows to link related contexts, by explicitly associating them through the common concepts they share, even when they are expressed in different ways. Therefore, we extended the ontology previously suggested so that all contextual values (not only the concrete ones) are classified in it. In this way we formalize the contexts definitions and representations, regulating their use. Our ontology is based on Wordnet (http://wordnet.princeton.edu), which is both a terminology and a constitutive ontology that supports multilanguages and implements a set of semantic relations between concepts (e.g., synonymy, part-set, etc.). WordNet must be extended since it only provides a basic terminology and a first group of semantic relations. However, the definition of relations specific to the contextual aspects is required to obtain an ontology of contexts. To this aim we devise a domain ontology, which includes concepts and basic relations for sensors and contexts, and can be extended by more specific ontologies. For example, to define the concepts related to "house" we both refer to the basic relations and extend the generic concepts from a previously defined "building" ontology.

A fourth problem concerns subjectivity: since a "one-size-fits-all" approach will most likely be inadequate for users with different needs, we have introduced some other functionalities that allow users to personalize their inferential system. A user can associate to each context of interest in the network two values called *privacy* and *importance*. These values refer to two thresholds managed by the user. The first one refers to the sensibility of the contextual information; contexts with privacy value higher than the privacy threshold will not be diffused outside the mobile device. The importance value acts somehow in the opposite direction: it allows a contexts to be made public to remote servers even if its probability is lower than the filtering threshold. Moreover, the user has the possibility to modify and personalize the network by adding or removing nodes (contexts) and changing the probability relations between contexts. This feature is crucial in order to make the network more suitable to model the user's current situation.

4 IMPLEMENTATION

In order to verify the feasibility of our approach, we concretely developed a simple prototype based on the above mentioned ideas. Rather than developing a complete system, we focused our efforts only on the implementation of our inferential approach: thus, the sensors information is simulated by an external application, in order to avoid low level details and to concentrate ourselves just on the context inferences.

More precisely, the inferential system is composed by networks and an inferential engine. The latter in particular manages the acquisition of contextual values from sensors, the acquisition of networks and the execution of the inferences on them. This system has been implemented in Java using the package JavaBayes (www.cs.cmu.edu/javabayes) for Bayesian networks and JESS (http://herzberg.ca.sandia.gov/) for the rule-based stage. The Bayesian networks have been designed with the editor BNJ (http://bndev.sourceforge.net) and saved in XML format.

Two distinct environments have been chosen as application candidates of our prototype: a domotics environment, with an inferential network that manages home contexts,and an automotive environment with two networks, one for the car and one for the highway.

4.1 Domotics environment

The domestic environment is limited enough to avoid complexity issues and to be explicitly managed but, at the same time, it presents a quite heterogeneous set of situations. Also, it is well known and the definition of the rules and the relations between contextual values in the Bayesian network and their probabilities can be generated taking as examples our everyday life. For this prototype we take into account the following concrete contextual information (which we consider the most meaningful in the domestic field): user location within the house, time of the day, user movement, light level, sound level, temperature, and humidity.

In this prototype, starting from the concrete contexts, the system infers the abstract contexts. For instance, when a user is in his home, his device will receive an application to control the domotics system; when a user is watching TV his device will turn into a TV remote; when the user is having a shower, his device will play a list of mp3. Also, context —and application— filtering can be performed on the basis of the probability value. For example, if the user could define an 80% threshold, to discard all the contexts with a lower probability to be discarded, and to retrieve the most suitable applications or web pages (MoBeLets) on the basis of the higher probability contexts. Then, knowing that "the user is in kitchen" and "it is lunch time" and "the user is not in movement", the system can infer with a certain probability that "the user is having lunch", and with a lower probability "the user is preparing lunch".

It is important to observe that a higher number of managed concrete contextual values and relative details corresponds to a more probabilistically correct inference of the abstract contexts. For instance, it is possible to infer the abstract context "user is having a shower" from the concrete context "user in bathroom". However, increasing the concrete contexts number, managing also information like humidity and sound level, the system can provide a more correct representation of the current user context. Anyway, the set of concrete contextual values taken into consideration is more than enough for the purposes of this prototype.

4.2 Automotive environment

The second environment is derived from the automotive field. In this case we have used two different networks. The first one is related to the car: it receives concrete context values from car sensors (e.g., oil and water condition, etc.) and infers the current car status. The second network models a highway, and it uses several concrete contexts: weather, location, traffic information, speed limits, etc.

When the user enters his car, his mobile device receives the car network; when he drives into the highway, the device receives the highway network, and the system integrates it with the other networks (the car network, in this example). As the networks are obtained accordingly to the user location, they are integrated on the basis of this contextual values: the more general network receives in input the contexts inferred by the less general network. In this prototype the highway network receives in input the contexts inferred by the car network. This integration allows a more precise description of user's current context: for example, the "user at gas station" context, inferred by the highway network, acquires different meanings if the "car broken" context has been inferred by the car network.

Figure 3 shows two screenshots of our prototypes.

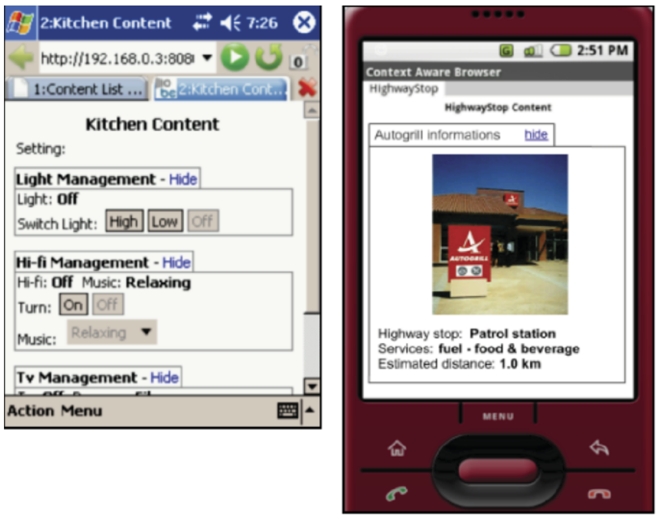
Kitchen Content
Setting:
Light Management
Light: Off
Hi-fi Management
Hi-fi: Off Music: Relaxing
Highway stop: Patrol station
Services: fuel - food & beverage
Estimated distance: 1.0 km

Figure 3. Home and highway examples (Windows Mobile / Android).

5 CONCLUSIONS AND FUTURE WORK

In this paper we have presented a new approach to context inference in context-aware applications on mobile devices. In particular we have proposed an approach based on the combination of rule-based systems, Bayesian networks, and ontologies; we have shown how these three AI tools (together with multiagent systems) can be exploited in real-world context aware systems. Differently from past works in the context-aware field, we have not focused only on particular contextual values, as our inferential infrastructure is able to work in a more general way: our aim was to combine contexts to determine new, more abstract contexts. The feasibility of our approach has been demonstrated through a concrete application within MoBe, a framework for context-aware applications.

The work presented is just a preliminary work. Although it demonstrates the feasibility of our approach, several questions have to be answered. The following step concerns a more robust and complete implementation of our prototype in order to execute a complete user testing. Moreover we are going to study and integrate in our system the *context history*, i.e., the set of all inferred contexts, organized in a temporal order. We are going to investigate how compute, starting from the history, significant statistics on the user's contexts, that can be useful to dynamically adapt the inferential network to the user experience.

References

[1] G. D. Abowd, A. K. Dey, P. J. Brown, N. Davies, M. Smith, and P. Steggles. Towards a better understanding of context and context-awareness. In *HUC '99: Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing*, pages 304–307. Springer-Verlag, 1999.

[2] J. Bacon, J. Bates, and D. Halls. Location-oriented multimedia. *IEEE Personal Communications*, 4(5):48–57, ottobre 1997.

[3] M. Bazire and P. Brezillon. Understanding context before using it. In *Proceedings of CONTEXT 2005*, pages 29–40. Springer-Verlag, luglio 2005.

[4] G. Biegel and V. Cahill. A framework for developing mobile, context-aware applications. In *Proceedings of Second IEEE International Conference on Pervasive Computing and Communication, PerCom 2004*, marzo 2004.

[5] P. J. Brown, J. D. Bovey, and C. Xian. Context-aware applications: from the laboratory to the marketplace. *IEEE Personal Communications*, 4(5):58–64, 1997.

[6] G. Chen and D. Kotz. A survey of context-aware mobile computing research. Technical Report TR2000-381, Dept. of Computer Science, Dartmouth College, novembre 2000.

[7] P. Coppola, V. Della Mea, L. Di Gaspero, S. Mizzaro, I. Scagnetto, A. Selva, L. Vassena, and P. Zandegiacomo Riziò. MoBe: A framework for context-aware mobile applications. In *Proceedings of Workshop on Context Awareness for Proactive Systems CAPS 2005*, pages 55–65. HIT, giugno 2005.

[8] C. di Flora, O. Riva, K. Raatikainen, and S. Russo. Supporting mobile context-aware applications through a modular service infrastructure. In *Proceedings on the Sixth International Conference on Ubiquitous Computing, UbiComp 2004*, 2004.

[9] P. Korpipää, M. Koskinen, J. Peltola, S.-M. Makela, and T. Seppanen. Bayesian approach to sensor-based context awareness. *Personal Ubiquitous Comput.*, 7(2):113–124, 2003.

[10] A. Newell and H. A. Simon. *Computer Augmentation of Human Reasoning*. Spartan Books, 1965.

[11] J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Publishers, Inc., 1988.

[12] B. Schilit and M. Theimer. Disseminating active map information to mobile hosts. *IEEE Network*, 8(5):22–32, 1994.

[13] A. Schmidt, K. A. Aidoo, A. Takaluoma, U. Tuomela, K. V. Laerhoven, and W. V. de Velde. Advanced interaction in context. In *Proceedings of First International Symposium on Handheld and Ubiquitous Computing (HUC'99)*, pages 89–101. Springer-Verlag, 1999.

[14] D. West, T. Apted, and A. Quigley. A context inference and multi-modal approach to mobile information access. In *AIMS 2004, Artificial Intelligence in Mobile Systems 2004 (in conjunction with UbiComp 2004)*, 2004.

[15] T. Zhang. An architecture for building customizable context-aware applications by end-users. In *Proceedings on Second International Conference, PERVASIVE 2004*, aprile 2004.