ELEKTROTEHNIŠKI VESTNIK 79(1-2): 41-46, 2012
ENGLISH EDITION

Taxonomy of context-aware systems

Rok Žontar, Marjan Heričko, Ivan Rozman

University of Maribor, Faculty of Electrical Engineering and Computer Science, Smetanova 17, 2000 Maribor, Slovenia
E-mail: rok.zontar@uni-mb.si

Abstract. In this paper we present conclusions, drawn from our research on context-aware systems. Based on reviewing the literature, from which we collected definitions of context and context awareness. We provide the basis for a taxonomy of context-aware systems grouped in eight categories. They include technical aspects of these systems as well as some views of privacy and security. Usefulness of taxonomy is demonstrated with a classification example based on the taxonomy categories.

Keywords: context-aware systems, sensors, taxonomy, classification

1 INTRODUCTION

The massive evolution of mobile devices like smart phones and ubiquitous information solutions have led to a new complex environment in which users expect useful services and information. But the extent of data and services continues to grow at an immense speed. It has become so vast that users are having difficulties and spend much time choosing the appropriate services. This inspired the idea of using context for filtering out data and providing services tailored to users.

Systems which take advantage of context are called context-aware systems and they enable new development opportunities for service providers and their end users. By collecting and processing context data, the systems behavior can be adjusted and its efficiency in searching for information and services increased. This leads directly to a higher user satisfaction. Combined with mobile devices, these mechanisms have proven to be particularly effective and are experiencing a widespread use on different mobile application frameworks.

Results of our reviewing the current literature and summarized achievements of related works provided the basis for analyzing context-aware systems. Despite the many common characteristics, these systems differ in operation, architecture style and use of context. To achieve a higher level of systematization, we decided to prepare taxonomy of the context-aware systems presented in this paper.

The paper is organized as follows. After a short introduction, in chapter 2 we present some of the definitions of context and context-aware computing. In chapter 3 we describe some of the basic characteristics of context-aware systems. In chapter 4 we first present the taxonomy of the context-aware systems and then a classification of context-aware systems, based on the presented taxonomy. The paper ends with a discussion on our research plans for our future work.

2 DEFINITION OF CONTEXT AND CONTEXT-AWARENESS

To get a better insight into the idea context, we first looked at some of its definitions used in the literature. The term was first introduced by Schilit and Theimer [1] who defined it as *a location or identity of people and objects in close surroundings*. Similar definitions were provided by Brown et al. [2] and Ryan et al. [3]. Although such definitions are quite common, they are difficult to generalize in practice. That is why researchers tend to use a more general definition. A notable effort in this direction was made by Dey and Abowd [4] who tried to unify the view on context and context-aware applications.

If context is used in information services rather than mobile devices or pervasive environments, the definition by Doulkeridis et al. [5] is more suitable. They distinguish two kinds of context, namely the client and service context. In the execution process, these contexts are matched in order to retrieve more relevant results.

There is not much difference between the definitions of context and context-aware computing. It is therefore not surprising that the first definitions of context-awareness were proposed by Schilit and Theimer [1]. They started by defining applications as context-aware if they were acquainted with context, but this was later refined by Dey and Abowd [4].

The truth is that there has been no common agreement reached in understanding what context really is. Although many authors use the term context, they fail to precisely define what it means exactly. Also, authors tend to perceive context quite differently. In our

Received April 11, 2012
Accepted April 23, 2012

research, context was understood as a collection of additional data about the user allowing a computer program to filter out relevant data or services based on its requirements.

3 CONTEXT-AWARE SYSTEMS

Over time, many frameworks and applications have been developed that can be sorted into groups of context-aware systems. Generally, they differ in the provided functionalities, naming and positioning of architectural layers and other architectural concepts. In order to develop a meaningful taxonomy, we first investigated some common characteristics which had been highlighted by other authors before us (Baldauf et al. [6] and Lee et al. [7]). Based on their work and the presented meta-architectures, we will be able to analyze and elaborate context-aware systems.

To handle presentation, management, reasoning and analysis of context data, commonly complex architectures and many subsystems of the context-aware systems are needed. It is only through a well-coordinated cooperation of all these subsystems that make it is possible to achieve certain functionalities. Although there are many different systems, they all follow four basic steps to manage context [7]. These are:

- context acquisition,
- context storage,
- context abstraction and
- use of context data.

The first step in acquiring raw context data is the use of physical or virtual sensors. When they are gathered, the systems usually store them in a repository. When doing so, the data must be organized in a data or context model. The next step is optional, because only some systems interpret and aggregate context data. This is called context abstraction. The last step in this process is the use of the raw or abstracted context data in applications or services.

4 TAXONOMY

In this section we will present and describe categories of our taxonomy. We will identify all the criteria and explain why we decided to include them.

4.1 System Types

Similarly as any other form of software, context-aware systems too, can be implemented in different ways. Early systems were implemented as stand-alone applications running on a particular device or system. Because of this, they were adapted to specific sensors and other types of hardware resulting in negative consequences on transferability and popularity of use. It is for this reason that new systems have been developed in the form of frameworks or middleware. While the reuse capability of the first is higher, the latter focuses primarily on the use of context in web services and service-oriented architectures [7]. Examples of the context-aware frameworks are the Context Toolkit [8], the Hydrogen framework [9] and CoBrA [10]. On the other hand, Service-Oriented Context-Aware

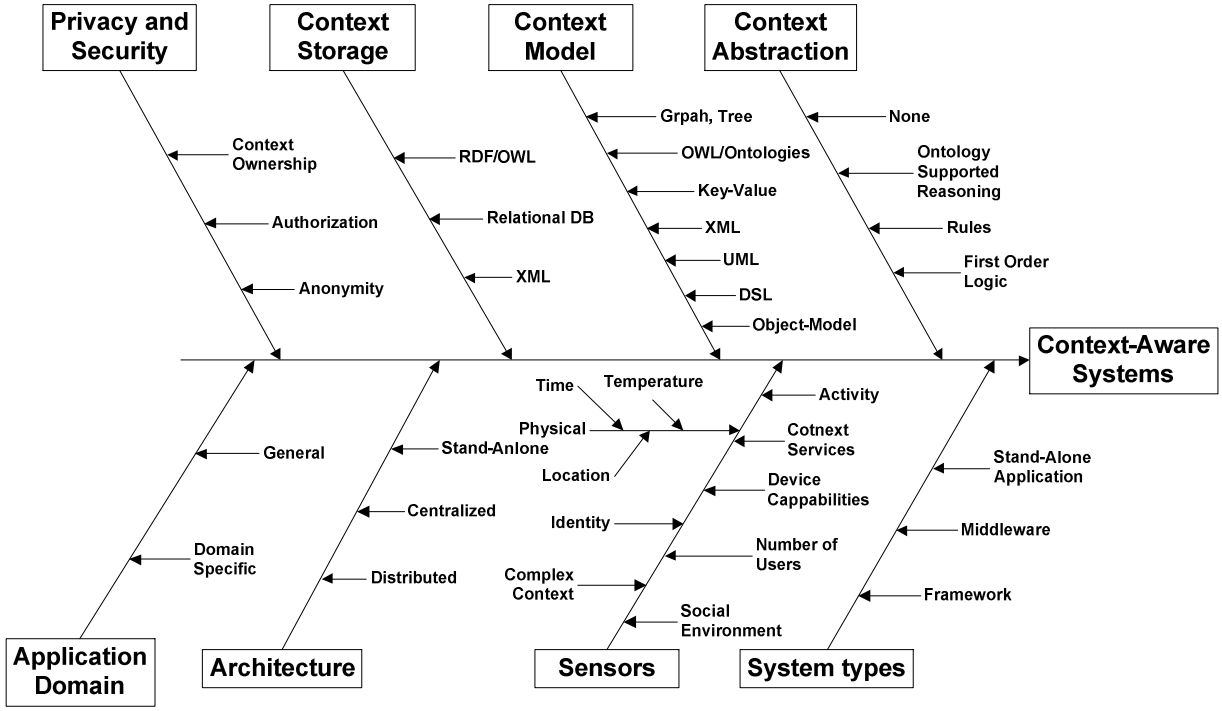

Figure 1. Taxonomy of context-aware systems

Middleware – SOCAM [11] and Gaia [12] are popular representatives of the middleware approach.

4.2 Application Domain

The application domain is closely bound to the type of the system involved. While all the context-aware systems implemented as stand-alone applications, are domain specific, the others show a clear and noticeable trend towards domain independence [7]. Thus, it is not surprising that it follows the development of object-oriented systems with a high level of reuse and a trend towards the use of frameworks.

4.3 Architecture

There are three categories of architecture styles of context-aware systems: stand-alone, distributed and centralized. The simplest is the stand-alone architecture because the application has direct access to all sensors and does not allow for sharing context between different devices. The advantage of this kind of architecture is in being quickly developable, but on the other hand, it is limited only to self-gathered context data. Therefore it is suitable only for small and simple domain-specific applications.

A more advanced version is the distributed architecture. Context-aware applications implemented in this way can store context data on many remote devices, which means that a central server is not necessary. Each device is independent and thus the malfunction or loss of a device does not have a huge impact on the system as a whole. The weak points of this approach are the communication links between the devices themselves for being often implemented as ad-hoc solutions, which are difficult to control and maintain [7]. Mobile devices lack the computing resources and power for such a form of implementation. Therefore, this kind of implementation is not suitable for applications that are in high demand of the processing power or other resources.

In a centralized architecture, all sensors and devices are connected to a central server, which is often called the context server. It has all the necessary storage and computing capacity available. Therefore, all context data is stored there. When a device requires additional data, a query to the context server is sent, which in return sends back a result. In this architecture style, all communication links lead to the context server, which allows them to be simple and common. Its direct result is that adding and exchanging devices and sensors in such a system are very simple.

4.4 Sensors

Sensors play a very crucial role in the context-aware systems, because they acquire context data. Some of the sensors were already discussed in chapter 3, where we explained that sensors can be categorized as physical or virtual, or a combination of both. In this work, physical sensors are the ones most commonly mentioned. These sensor types especially include location [4,5,13,14,15,16,17], followed by time [4,5,14,17] and temperature [13]. Some other types of sensors also include:

- user identity,
- user activity,
- device capability,
- number of users,
- social environment,
- context services and
- complex context.

Although it is not possible to detect the user identity or the user profile with a sensor, it is right to list them in this category because of their important impact on the behavior of the system which has been proven in a number of researches [5,15,16,17]. A user can directly affect the behavior of a system by setting preferences in her/his user profile. It can also happen that a user steps over the context border and becomes the central part of a system. In [18] a system suggests ways of communication with a person, based her/his preferences, location and activity. A similar effect can also be seen in the number of people or the social environment in which a person resides [14].

Device capabilities are closely related to the provided content. In [5], it is shown how device capabilities can be used to affect search results. They identify the screen size and supported media types as the main factors that can be used to filter out multimedia content that is only suitable for some kinds of devices.

As already mentioned context services present an abstraction of sensors and can be used with the acquisition of all kinds of context data. Recent context-aware systems often use them as an additional layer of abstraction. Complex context, as presented in [4], is a synthesis of different context information, which by themselves would not provide the same degree of information as they do when combined together.

4.5 Context Model

A context model is used to store and process context data. Choosing and developing the model is a difficult task, because it has to cover as many potential uses as possible [19]. Baldauf et al. [6] review the most relevant approaches to context modeling. They are based on data structures used to present and exchange context data. They distinguish between the following six categories:

- key-value models,
- mark-up scheme models,
- graphical models,
- object-oriented models,
- logic-based models and
- ontology-supported models.

Having reviewed existing literature enabled us to get an insight into the models of context-aware frameworks

and applications. This is why we decided to expand these categories. We added a graph or a tree model described in [5] and the domain specific language (DSL) from [19]. Finding no model that would be based on logic, we decided to merge it with the ontology-supported model group.

Furthermore, we would also like to mention ContextUML [13]. This is a Unified Modeling Language (UML) based notation for describing context data. In recent literature, a trend is noticed towards using ontologies as a model of context data. Ontologies provide a way of describing concepts and the relationships between them. Because of their high degree of expressiveness and the ability to use reasoning on them, they are regarded as a very prominent field of technology in this area.

4.6 Context Abstraction

Context data can be acquired directly from the sensors, but only the use of data abstraction leads to new and more useful results. As said above, there are many ways to model context. It is reasonable that the choice of the model reflects the ability to use context abstraction in a system. When context data is described by using ontologies, abstraction takes place with a semantic approach using ontology-supported reasoning.

In other context models, rules or the first-order logic can be applied for abstraction. But the reasoning ability in these cases is dramatically limited.

4.7 Context Storage

Some context-aware systems keep their data in a repository. This enables them to use the data at runtime or access them at a later point in time, when they can have a backward effect on the then-current context. But the ways of storing data are different and strongly connected with the selected context model. Most commonly, relational databases are used for providing a standardized and platform-independent way to access and store data. Serialization of the context data to XML (Extensible Markup Language) is also often used, mainly to distribute context. With the rise of ontologies, a growth of storage in RDF[1] (Resource Description Framework) documents and triple stores has been noted. Context-aware systems, which take advantage of the service architectures, often use context services for persisting context data.

4.8 Privacy and Security

The privacy of data and its security is a very important aspect of context-aware systems. Continuous tracking of users and their profiles may cause fear and anxiety among users, which can result in the rejection of certain applications or devices. It is therefore necessary to incorporate suitable security measures for protecting and limiting access to this data. This is particularly important in systems storing context data in a lasting form. Here, the distinction should be made between the classical security measures of securing data in information systems and the data security of context-aware applications. The point is that the following questions regarding the degree of freedom in accessing context data should be answered. Is location information really necessary? Should access to a user profile be allowed? How is the history recorded? All these questions as well as many others need to be answered in order to provide a trustworthy context-aware system.

But the lack of research in this area is plainly apparent. Only few authors mention security mechanisms for protecting the users' privacy. In most cases, they only focus on the architecture and functionalities of their systems. Langheinrich [20] emphasizes the risk that may occur to users of ubiquitous mobile solutions. He also highlights the need to inform the user about what information is being collected and also obtain the user's approval for doing so. Furthermore, he underlines the importance of anonymity and concludes with some guidelines for data collection:

- Collect data only for a specific and well defined purpose;
- Collect only data relevant for that purpose (and not more);
- Keep only data for as long as it is necessary for that purpose;

One of the first researchers to mention privacy was Dey [8]. In his Context Toolkit, he implemented a simple access mechanism to secure the users' privacy in the form of context ownership. CoBrA [10] offers an ontology supported language (REI) enabling users to define rules in order to control the sharing of personal data.

5 CLASSIFICATION EXAMPLE

In order to evaluate our taxonomy, a classification example was made. We gathered some context-aware systems from the literature and classified them according to our taxonomy. The results are presented in Table 1.

Eight samples of context-aware applications have been selected to cover all types of system. The classification has been performed in strict compliance with the proposed taxonomy. For all categories where only one choice is possible, the precise one was selected. For the other categories all supported or mentioned elements are listed for each system. If no element is applicable it is denoted by a "/" sign.

[1] http://www.w3.org/RDF/

Table 1. Classification of context-aware systems made according to our taxonomy

Context-aware system	System Types	Application Domain	Architecture	Sensors	Context Model	Context Abstraction	Context Storage	Privacy and Security
AMS [1]	Stand-alone application	Domain specific	Stand-alone	Location	/	/	/	/
Context Toolkit [8]	Framework	General	Distributed	Location, identity, time, activity	Key-Value	Rules	/	Authorization
Hydrogen [9]	Framework	General	Distributed	Location, identity, time	Object-model	/	/	/
Gaia [12]	Framework	General	Distributed	Location	Key-Value	First order logic	XML	/
CoBrA [10]	Framework	General	Centralized	Location, activity	Ontology	Ontology supported reasoning	RDF/OWL	Authorization
SOCAM [11]	Middleware	General	Centralized	Context services	Ontology	/	RDF/OWL	/
Enhanced CoCA [21]	Framework	General	Distributed	Location, activity	Ontology	First order logic	Relational database	/
Context Management Framework [22]	Framework	General	Centralized	Location, time, temperature, activity, device capability	Ontology	Ontology supported reasoning	XML	/

6 CONCLUSION

Following results of our research, we provide an overview of context and context-aware computer systems. We classify what context is and how it is defined by other researchers. We answer the question of how the term context is understood and in what are the benefits from using it in computer systems. The answer to these and similar questions is found by reviewing existing literature, where system architecture and currently used technologies were at the forefront of our attention.

To better understand context, we first examined some of its definitions. Despite having a rather common understanding of context, the authors express it in many different ways. Probably the most agreed-upon definition is given by Day and Abowd in [4]. Our attention is then devoted to systems built by using context. We sum up some of the common architectural characteristics and describe the process of acquiring, processing and using context. Based on the acquired knowledge, we create a taxonomy of context-aware systems to set up a systematic approach to context-aware systems. Grouping them in eight categories, where each system is accurately classified. To gain a better overview, the taxonomy is presented in a cause-effect diagram.

Our conclusion is that the area of context-aware systems is not sufficiently researched. Further work can be done in any of the eight categories. Probably the most unresearched area is context acquisition. The current systems often promote the support of multiple sensors but, in practice, only location sensing has been implemented. Intelligent mechanisms are needed to acquire other types of context. Other areas needing to be researched are context storage and procession as well as privacy and security for which there has been almost no research work done so far.

REFERENCES

[1] B. Schilit and M. Theimer, "Disseminating Active Map Information to Mobile Hosts," IEEE Network, vol. 8, no. 5, pp. 22-32, 1994.

[2] P. J. Brown, J. D. Bovey, and X. Chen, "Context-Aware Applications: From the Laboratory to the Marketplace," IEEE Personal Communications, vol. 4, no. 5, pp. 58-64, 1997.

[3] N. Ryan, J. Pascoe, and D. Morse, "Enhanced Reality Fieldwork: the Context-Aware Archaeological Assistant," in In Computer Applications in Archaeology, 1997.

[4] Anind K. Dey and Gregory D. Abowd, "Towards a Better Understanding of Context and Context-Awareness," in HUC '99: Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing, 1999.

[5] Christos Doulkeridis, Nikos Loutas, and Michalis Vazirgiannis, "A System Architecture for Context-Aware Service Discovery," Electronic Notes in Theoretical Computer Science, vol. 146, no. 1, pp. 101-116, 2006.

[6] Matthias Baldauf, Schahram Dustdar, and Florian Rosenberg, "A survey on context-aware systems," International Journal of Ad Hoc and Ubiquitous Computing, vol. 2, no. 4, pp. 263-277, 2007.

[7] Sankeun Lee, Juno Chang, and Sang-goo Lee, "Survey and Trend Analysis of Context-Aware Systems," Information-An International Interdisciplinary Journal, vol. 14, no. 2, pp. 527-548, 2011.

[8] Anind K. Dey, "Understanding and Using Context," Personal and Ubiquitous Computing , vol. 5, no. 1, pp. 4-7, 2001.

[9] T. Hofer et al., "Context-awareness on mobile devices - the hydrogen approach," in Proceedings of the 36th Annual Hawaii International Conference on System Sciences, 2003, pp. 6-9.

[10] H. Chen, T. Finin, and A Joshi, "An ontology for context-aware pervasive computing environments," The Knowledge Engineering Review, vol. 3, no. 18, pp. 197-207, 2003.

[11] T. Gu, H.K. Pung, and D.Q. Zhang, "A middleware for building context-aware mobile services," in Proceedings of IEEE Vehicular Technology Conference (VTC), Milan, 2004.

[12] Manuel Román, Christopher Hess, Renato Cerqueira, Roy H. Campbell, and Klara Nahrstedt, "Gaia: A Middleware Infrastructure to Enable Active Spaces," IEEE Pervasive Computing , vol. 1, pp. 74--83, 2002.

[13] Quan Z. Sheng and Boualem Benatallah, "ContextUML: A UML-Based Modeling Language for Model-Driven Development of Context-Aware Web Services Development," in Proceedings of the International Conference on Mobile Business (ICMB '05), Washington, DC, 2005, pp. 206-212.

[14] Albrecht Schmidt, "Implicit Human Computer Interaction Through Context," in 2nd Workshop on Human Computer Interaction with Mobile Devices, Edinburgh, 1999.

[15] Stephan Reiff-Marganiec and Hong Qing Yu, "Automated Context-aware Service Selection for Collaborative Systems," in CAiSE '09 Proceedings of the 21st International Conference on Advanced Information Systems Engineering, Amsterdam, 2009, p. 261.

[16] Keita Fujii and Tatsuya Suda, "Semantics-based context-aware dynamic service composition," ACM Transactions on Autonomous and Adaptive Systems, vol. 4, no. 2, Maj 2009.

[17] Nasser Ghadiri, Mohammad Ali Nematbakhsh, Ahmad Baraani-Dastjerdi, and Nasser Ghasem-Aghaee, "A Context-Aware Service Discovery Framework Based on Human Needs Model," Lecture Notes in Computer Science, pp. 404-409, 2007.

[18] Jiehan Zhou and Jukka Riekki, "Context-Aware Pervasive Service Composition," in 2010 International Conference on Intelligent Systems, Modelling and Simulation, Liverpool, 2010, pp. 437-442.

[19] Karen Henricksen and Jadwiga Indulska, "Developing context-aware pervasive computing applications: Models and approach," Pervasive and Mobile Computing, vol. 2, no. 1, pp. 37-64, 2006.

[20] Marc Langheinrich, "Privacy by Design - Principles of Privacy-Aware Ubiquitous Systems," in UbiComp '01 Proceedings of the 3rd international conference on Ubiquitous Computing , London, 2001, pp. 273-291.

[21] Dejene Ejigu, Marian Scuturici, and Lionel Brunie, "CoCA: A Collaborative Context-Aware Service Platform for Pervasive Computing," in Fourth International Conference on Information Technology, INT'07, Las Vega, NV, 2007, pp. 297-302.

[22] Panu Korpipaa, Jani Mantyjarvi, Juha Kela, Heikki Keranen, and Esko-Juhani Malm, "Managing context information in mobile devices," IEEE Pervasive Computing, vol. 2, no. 3, pp. 42-51, July-September 2003.

Rok Žontar graduated in 2009 from the Faculty of Electrical Engineering and Computer Science, University of Maribor, where he is employed as a researcher of the National Young-researcher Scheme. His research interests include context-aware systems, model-driven architectures, semantic web and ontology development and similar.

Marjan Heričko received his BSc, MSc and PhD from the Faculty of Electrical Engineering and Computer Science, University of Maribor in 1989, 1993 and 1998, respectively. He is currently a full professor at the same faculty.

Ivan Rozman received his BSc, MSc and PhD from the Higher Technical School in Maribor in 1977, 1980 and 1983, respectively. He is a full professor and the head of the Institute of Informatics at the Faculty of Electrical Engineering and Computer Science, University of Maribor.