An Ontology-based Conceptual Model for Composing Context-Aware Applications

Eleni Christopoulou, Christos Goumopoulos, Ioannis Zaharakis and Achilles Kameas
Research Academic Computer Technology Institute
Research Unit 3, Design of Ambient Intelligent Systems Group
61 Riga Feraiou str. 26221, Patras, Greece
+30 2610 273496
{hristope, goumop, jzaharak, kameas}@cti.gr

ABSTRACT

Ubiquitous computing (UbiComp) applications operate within an extremely dynamic and heterogeneous environment. Thus context definition, representation, management and use become important factors that affect their operation. UbiComp applications have to dynamically adapt to changes in their environment as a result of users' or other actors' activities. To ease the development of such applications it is necessary to decouple application composition from context acquisition and representation, and at the same time provide universal models and mechanisms to manage context. This paper presents experiences with using an ontology to represent context of operation together with decision making for UbiComp applications that result from the composition of functionally independent components. These components were embedded in everyday objects, hence (a) their services were affected by their physical properties, (b) their context of operation was defined by the existence / availability of the objects, and (c) their collective functionality was emerging from a set of interactions among them.

Keywords
Ubiquitous computing, ontologies, context-awareness

INTRODUCTION

The vision of Ambient Intelligence (AmI) implies a seamless environment of computing, advanced networking technology and specific interfaces [16], [14]. Technology becomes embedded in everyday objects and environments such as furniture, clothes, vehicles, roads and smart materials, and people are provided with the tools and the processes that are necessary in order to achieve relaxing interactions with this environment. The AmI environment can be considered to host several Ubiquitous Computing (UbiComp) applications, which make use of the infrastructure provided by the environment and the services provided by the objects therein.

Since the UbiComp applications operate within an extremely dynamic and heterogeneous environment, the context definition, representation, management and use become important factors that affect their operation. UbiComp applications have to dynamically adapt to changes in their environment as a result of users' or other actors' activities. To ease the development of such applications it is necessary to decouple application composition from context acquisition and representation, and at the same time provide universal models and mechanisms to manage context.

According to our approach UbiComp applications result from the composition of functionally independent components. These components were embedded in everyday objects, hence (a) their services were affected by their physical properties, (b) their context of operation was defined by the existence / availability of the objects, and (c) their collective functionality was emerging from a set of interactions among them. The target of this paper is to present our experience with designing and using an ontology to compose context-aware UbiComp applications and our proposition for an ontology-based context modeling, management and reasoning process.

The rest of the paper is organised as follows. First we describe how context is modelled and used in various UbiComp applications along with the definition and use of context in our approach. Then we present the structure and the content of the ontology that we developed and integrated into a middleware that supports the composition of context-aware UbiComp applications follow. Finally we conclude with our proposition for ontology-based context modeling, management and reasoning process in UbiComp applications and our vision for the use of context in future UbiComp environments.

CONTEXT-AWARE UBICOMP APPLICATIONS

According to [4] context is: "Any information that can be used to characterise the situation of entities (i.e. whether a person, place or object) that are considered relevant to the interaction between a user and an application, including the user and the application themselves. Context is typically the location, identity and state of people, groups and computation and physical objects." In UbiComp applications different kinds of context can be used like physical information, e.g. location and time, environmental information, e.g. weather and light, personal information, e.g. mood and activity. Though, the term context is mostly referred to information relative to location, time, identity and spatial relationships.

A number of informal and formal context models have been proposed in various UbiComp systems. Among systems with informal context models, Context Toolkit [5] represents context in form of attribute-value tuples, and Cooltown [9] proposed a Web based model of context in which each object has a corresponding Web description. Both ER and UML models are used for the formal context model presented in [7]. In Gaia system [11], [12] the context is represented as first-order predicates written in DAML+OIL. Two similar approaches are COBRA-ONT [2], an ontology for context-aware pervasive computing environments represented in OWL, and CONON, [13] an OWL encoded context ontology for pervasive computing environments.

It has been acknowledged from many researchers that it is a necessity to decouple the process of context acquisition and interpretation from its actual use, by introducing a consistent, reliable and secure context framework which can facilitate the development of context-aware applications [1], [5], [12], [10], [7]. In this respect, we have proposed a high-level conceptual model, the plug/synapse model, which eases the composition of context-aware UbiComp applications and separates it from the process of context acquisition.

The key idea behind this conceptual model is that artefacts can be treated as components of a UbiComp application and users can compose UbiComp applications simply by creating associations between the artefacts. The plug/synapse model serves as a common interfacing mechanism between artefacts providing the means to create large scale systems based on simple building blocks. Plugs make visible the artefact's properties, capabilities and services to people and to other artefacts, while synapses are associations between two compatible plugs.

In terms of the application developer, the plugs can be considered as context-providers that offer high-level abstractions for accessing context (e.g., location, state, activity, etc.). In terms of the service infrastructure (middleware), they comprise reusable building blocks for context rendering that can be used or ignored depending on the application needs. Each context-provider component reads input sensor data related to the specific application and can output either low level context information such as location, time, light level, temperature, proximity, motion, blood pressure or high-level context information such as activity, environment and mood. An artefact from its own experience and use has two different levels of context; the low level is information acquired from its own sensors and the high level that is an interpretation of its low level context information. Additionally an artefact can get context information from the plugs of other artefacts; this context can be considered as information from a "third-person experience".

The application developers by establishing synapses between plugs both denote their preferences and needs and define the emerging behavior of the UbiComp application. From the service infrastructure perspective, the synapses determine the context of operation for each artefact; thus each artefact's functionality is adapted to the UbiComp application's structure.

Central to our approach is the use of an ontology, which provides a formal definition of the domain under consideration and a conceptual description of the domain knowledge. The next section presents our experience with designing and using such an ontology for a UbiComp system.

AN ONTOLOGY-BASED CONCEPTUAL MODEL

The GAS Ontology [3] that we developed conceptualises the Gadgetware Architectural Style (GAS) [8], which supports the composition of UbiComp applications from everyday physical objects enhanced with sensing, acting, processing and communication abilities. UbiComp applications are dynamic, distinguishable, functional configurations of associated artefacts, which communicate and/or collaborate in order to realize a collective behavior.

Designing the GAS Ontology

We have decided each artefact to have an ontology that contains the description of the basic concepts of UbiComp applications and their inter-relations. This knowledge must be common for all artefacts in order to support the feasible communication among them. Although an artefact's ontology should both describe the way that this artefact is used and represent its acquired knowledge. Specifically an artefact's ontology should represent its description and the descriptions of its plugs and services. Additionally an artefact's ontology should represent the new "knowledge" (experience) that the artefact has accumulated from the synapses that the artefact's plugs participate to. So the knowledge that each artefact's ontology represent cannot be the same for all artefacts, as it depends on the artefact's description and on the UbiComp applications that the artefact has participated in the past. Thus artifacts may end up having different ontologies.

Since artefacts' interoperability is designed to be based on their ontologies, the existence of different artefacts' ontologies could result to inefficient interoperability. In order to handle this issue we decided to allow each artefact to have a different ontology with the condition that all artefacts' ontologies will be based on a common vocabulary. According to this solution the GAS Ontology is divided into the following two layers: the GAS Core Ontology (GAS-CO) and the GAS Higher Ontology (GAS-

HO). Specifically the GAS-CO provides artefacts with the necessary common language that they need in order to describe their acquired knowledge, which is represented into the GAS-HO. Thus, although all artefacts have different knowledge, it is represented with terms and concepts common to all artefacts.

GAS Core Ontology (GAS-CO)

The GAS-CO describes the common language that artefacts use to communicate. So it must describe the semantics of the basic terms of UbiComp applications and define their inter-relations. Since a service discovery mechanism is necessary to a UbiComp system, this ontology must also contain a service classification to support this mechanism. An important feature of the GAS-CO is that it contains only the necessary information for the interoperability of artefacts in order to be very small and even artefacts with limited memory capacity may store it. The GAS-CO is static and it cannot be changed either from the manufacturer of an artefact or from a user. Following we describe the basic terms of the UbiComp applications and their inter-relations that are represented in the GAS-CO. A graphical representation is on Figure 1.

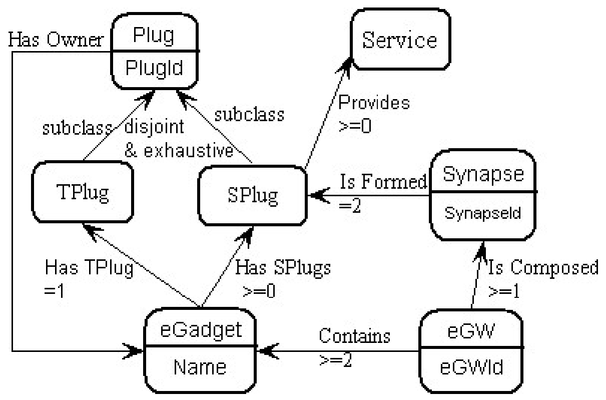

Figure 1: A Graphical Representation of GAS-CO

The core term of GAS is the eGadget (eGt); with this term we refer to artifacts of Ubicomp applications. In GAS-CO the eGt is represented as a class, which has a number of properties, like name etc. The notion of plug is represented in the GAS-CO as another class, which is divided into two disjoint subclasses; the TPlug and the SPlug. The TPlug describes the physical properties of the object that is used as an artifact like shape, material, etc.; note the cardinality restriction that an artifact must have exactly one TPlug. On the other hand an SPlug represents the artifact's capabilities and services; artifacts may have an arbitrary number of SPlugs. Another GAS-CO class is the synapse that represents the association between two plugs; a synapse may only appear among two SPlugs. Using the class of eGW the GAS-CO can describe the UbiComp applications that are created by the users; an eGW is represented by the artifacts that contains and the synapses that compose it. The class of eGW has two cardinality constraints; an eGW must contain at least two artifacts and a synapse must exist between their SPlugs.

The service classification

As an artefact through a plug provides a number of services, the GAS-CO contains a class for the notion of service. Since for the UbiComp applications a semantic service discovery mechanism is useful and the replacement of artefacts depends on the services that artefacts offer, a service classification is necessary.

In order to define such a service classification we first identified some services that various artefacts may offer; some results of this work are presented in Table 1. From these results it is clear that the services offered by artefacts depend on artefacts' physical characteristics and their sensors/actuators.

Artefact	Offered services
eLamp	switch on/off, light, heat
eBook	open/close, number of pages, current page
eDrawer	contains objects yes/no, number of objects, open/close, locked/unlocked
eMusicPlayer	sound, sound volume,, kind of music, play/pause/stop, next/previous track
eCarpet	object on it yes/no, objects' position, pressure, weight, frequency

Table 1: Services Offered By Artefacts

Next we had to decide how we should classify the services. The classification proposals that we elaborated are the following: by object category, by human senses and based on the signals that artefacts' sensors/actuators can perceive/transmit. We decided to combine these proposals so that to describe a more complete classification.

So we initially defined the following elementary forms of signals that are used: sonic, optic, thermal, electromagnetic, gravity and kinetic. These concepts are divided into lower level services (subclasses); e.g. the sonic service may be music, speech, environmental sound, and noise. Additionally services may have a set of properties; e.g. sonic can have as properties the volume, the balance, the duration, the tone, etc.

Finally we enriched this classification by adding services relevant to environmental information, like humidity and temperature, and the concepts of time, position and movement.

GAS Higher Ontology (GAS-HO)

The GAS-HO represents both the description of an artefact and its acquired knowledge. These descriptions follow the definitions contained in the GAS-CO. So, specifically the knowledge stored into the GAS-HO is represented as instances of the classes defined into the GAS-CO. For example the GAS-CO contains the definition of the concept SPlug, while the GAS-HO contains the description of a specific SPlug represented as an instance of the concept SPlug. Note that the GAS-HO is not a stand-alone ontology, as it does not contain the definition of its concepts and their relations.

Since the GAS-HO represents the private knowledge of each artefact, it is different for each artefact. Therefore we can envision GAS-HO as artefact's private ontology. Contrary to GAS-CO, the size of which is required to be small enough, the size of GAS-HO depends only on artefact's memory capacity. Obviously GAS-HO is not static and it can be changed over time without causing problems to artefacts communication. As the GAS-HO contains both static information about the artefact and dynamic information emerged from its knowledge and use, we decided to divide it into the GAS-HO-static and the GAS-HO-volatile.

The GAS-HO-static represents the description of an artefact containing information about artefact's plugs, the services that are provided through these plugs, its sensors and actuators, as well as its physical characteristics. The knowledge represented by the GAS-HO-static can only be updated if the physical or digital properties of the artefact change. So the information represented by the GAS-HO-static can be considered as a description of an artefact's "self".

On the other hand the GAS-HO-volatile contains information derived from the artefact's acquired knowledge and its use. Specifically it describes the synapses which the artefact's plugs are connected to, the UbiComp applications which it takes part to, as well as information about the capabilities of other artefacts that has acquainted through communication. An artefact's GAS-HO-volatile is updated during the artefact's various activities, like the establishment of a new synapse. Although the GAS-HO-static and the GAS-HO-volatile contain different knowledge, both of them are based on the GAS-CO and contain instances of its concepts.

Using the GAS Ontology

In this section we show how we have used the GAS Ontology, for the composition and deployment of context-aware UbiComp applications. The examples that follow are based on the "study" scenario. According to this scenario, two artifacts, an eBook and an eLamp, are associated via a synapse which is established between two plugs forming a "study" UbiComp application. When the user opens the eBook, the eLamp switches on, adjusting the light conditions to a specified luminosity level in order to satisfy the user's profile. Each artifact operates independently of the other. Two plugs are being used in this scenario: "open/close" reflecting the state of the book and "switch on/off" reflecting the state of the lamp.

The GAS Ontology is used in order to support the semantic interoperability among artefacts, to handle the necessary adaptivity of UbiComp applications and to provide them with context-awareness. Since all artefacts use the same vocabulary for the representation of their abilities the semantic interoperability among them is ensured. This common vocabulary is represented by the GAS-CO, so both the eBook and the eLamp artefacts have stored the same GAS-CO. The service classification is represented into the GAS-CO so that to support a service discovery mechanism. So, if a synapse is broken, e.g. because of an artefact's failure, another artefact that offers a similar semantically service will be found.

As the context information that is used in such UbiComp applications describes the physical and digital properties of artifacts, it is represented into both the GAS-CO and each artifact's GAS-HO-static. Specifically an artefact through its TPlug provides other artefacts and users with context information about its physical properties. On the other hand through its SPlugs, an artefact provides context information both low level, data from sensors, and high level, interpreted low level context. This kind of context information is represented into an artefact's GAS-HO-static by using the terms described into the GAS-CO.

Reportedly to the "study" scenario, the eLamp's GAS-HO-static contains information about eLamp's TPlug and "switch on/off" SPlug. Correspondingly the eBook's GAS-HO-static contains the description of eBook's TPlug and "open/close" SPlug. Specifically through its TPlug the eBook provides to the other artefacts and to people context information about its type, number of pages, dimensions and the sensors/actuators attached to it. The eLamp through its TPlug, provides similar information based on its own properties.

On the other hand, artefacts through their SPlugs provide different kind of context information than through their TPlugs. For example, the eBook's "open/close" SPlug reflects the state of the eBook; if it is open or close. We refer to such context information as low level context, since it represents raw data from a sensor. The information provided from an "open/close" SPlug that also notifies about the specific page on which the eBook is opened, is referred as high level context. Correspondingly, the eLamp's "switch on/off" SPlug informs the other artefacts and the people about the eLamp's state as well as about the eLamp's ability to provide the service of "light". The eLamp through this SPlug provides not only low level context information, like if it is switch on or off, but also high level context information, like the color of its light and the selected luminosity.

The GAS-HO-volatile of artifacts contains mainly knowledge emerged from the synapses that compose an UbiComp application. This information represents both context information from other artefacts ("third-person experience") and the artifact's behavior when it gets context information via its synapses. Note that these behaviors are defined by the developer of the UbiComp application. So when a user sets a synapse between two plugs, the knowledge emerged from this synapse is stored in the GAS-HO-volatile of both artifacts that participate to it.

According to the "study" scenario a synapse is established between the two plugs of eBook and eLamp. Both artefacts via this synapse can get context information, mentioned as "third-person experience", from the other artefact. For example, such "third-person experience" is the context information that the eLamp gets via this synapse about the eBook's state.

The description of this synapse is represented into both eBook's and eLamp's GAS-HO-volatile. Although this description is same for both artefacts, provides them with different "knowledge". Specifically for the eLamp this description defines its behavior on different context that it can get via this synapse; if it gets the information that the eBook is close then it must be switched off and if the eBook is open it must be switched on. On the other hand, since this synapse doesn't imply the transition of the eBook's state on the modifications of context, its description just denote that the eBooks' "open/close" SPlug participates to a synapse with an SPlug that provides the service "light". It is evident that the GAS Ontology contains both context information and the description of the behaviors in proportion to context; so it can support the context-awareness of such UbiComp applications.

CONCLUSIONS AND FUTURE WORK

Central to our approach is the use of an ontology, which provides a formal definition of the domain under consideration and a conceptual description of the domain knowledge. The GAS Ontology that we developed sufficiently supports the composition of context-aware UbiComp applications from everyday enhanced physical objects and it also address a number of key issues of such applications like adaptivity and semantic service discovery.

Although in order to enable the development of synergistic and scalable mixed communities of communicating artefacts and plants [6], we had to enhance the GAS Ontology incorporating a number of new concepts. The basic new concept that we integrated into the GAS Ontology is the ePlant. Similar to artefacts that are enhanced everyday objects, ePlants are plants enhanced with sensing, acting, processing and communication abilities. The description of an ePlant doesn't significantly differ to the one of an eGadget, as both have a TPlug and a set of SPlugs. Since their basic difference is based on their physical properties, which are represented by their TPlugs, these two concepts can be regarded as subconcepts of a higher concept. Thus we decided to add to our ontology the concept of eEntity, which is divided into the disjoint concepts of eGadget and ePlant.

The context model that we selected for these UbiComp applications is the same ontology-based model. Although we decided to use a context management and reasoning process especially for the definition of a plant's state and behaviour. The first step of the context management process is the acquisition of the low-level context; e.g. the plant signals from sensors. Then if the low-level context is not sufficient for an eEntity in order to make a decision, it will be interpreted to high-level context information. Having acquired its necessary context, an eEntity can assess its state and decide its appropriate reaction (behaviour).

The interpretation of high-level context information, the state assessment and the reaction selection will be based on a set of rules. For these UbiComp applications we will use rules and constraints (axioms) in operational representation forms, that provide inferential and validation mechanisms, so that to allow the use of the ontology for reasoning. The reasoning will be based on the definition of the ontology, which may use simple description logic or user-defined reasoning using first-order logic.

Our Vision

Future ubiquitous computing environments will involve hundreds of interacting and cooperating devices ranging from unsophisticated sensors to multi-form actuators. Although the majority of these devices may have limited resources (computation, memory, energy, etc) or may be only oriented to certain tasks, their collective behaviours through local interactions with their environment may cause coherent functional global patterns to emerge. Hence, the combination and cooperation of locally interacting artifacts with computing and effecting capabilities may trigger the continuous formation of new societies that provide services not existing initially in the individuals and exhibiting them in a consistent and fault-tolerant way.

As these societies are dynamically reconfigured aiming at the accomplishment of new or previous related tasks, their formation heavily depends not only on space and time but also on their context of previous local interactions, previous configured teams, successfully achieved goals or possibly failures. This means that in order to initially create, manage, communicate with, and reason about, such kinds of emergent ecologies, we need somehow to model and embed to these entities social memory, enhanced context memory, and shared experiences. One step to this end is the design and implementation of evolving multi-dimensional ontologies that will include both non-functional descriptions, and rules and constraints of application, as well as aspects of dynamic behaviour and interactions. A core ontology will be open and universally available and accessible; however, during the ecology life-time the core ontology is evolved into higher goal, application and context specific one. Hence, ontologies describing specific application domains can be proprietary.

REFERENCES

1. Biegel G., and Cahill V. A Framework for Developing Mobile, Context Aware Applications, *2nd IEEE Conference on Pervasive Computing and Communications*, Orlando, FL, March 14-17, 2004.
2. Chen, H., Finin, T., Joshi, A. An ontology for context aware pervasive computing environments, to appear, *Knowledge Engineering Review - Special Issue on Ontologies for Distributed Systems*, Cambridge University Press, 2004.
3. Christopoulou, E., Kameas, A. GAS Ontology: an ontology for collaboration among ubiquitous computing devices, to appear in *International Journal of Human – Computer Studies, Protégé special issue*, 2004.
4. Dey A. K. Providing Architectural Support for Building Context-Aware Applications, *PhD dissertation. Georgia Institute of Technology*, November 2000.Baldonado, M., Chang.
5. Dey A. K., Salber D. and Abowd G. D. A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications, *Human-Computer Interaction (HCI) Journal*, Volume 16 (2-4), pp. 97-166, 2001.
6. Goumopoulos, C., Christopoulou E., Drossos N. and Kameas A. The PLANTS System: Enabling Mixed Societies of Communicating Plants and Artefacts, to appear in *European Symposium on Ambient Intelligence (EUSAI)*, Eindhoven, the Netherlands, November 8-10, 2004.
7. Henricksen K., Indulska J., and Rakotonirainy A. Modeling Context Information in Pervasive Computing Systems, *In F. Mattern and M. Naghshineh, editors, Pervasive 2002*, pages 167– 180, Springer Verlag, Berlin, 2002.
8. Kameas A., et al. An Architecture that Treats Everyday Objects as Communicating Tangible Components, in the proceedings of the *1st IEEE International Conference on Pervasive Computing and Communications (PerCom03)*, Fort Worth, USA, 2003.
9. Kindberg T. et al. People, Places, Things: Web Presence for The Real World, *Technical Report HPL-2000-16, HP Labs*, 2000.
10. Nixon P. et al., Engineering context-aware enterprise systems, i*n Workshop on Engineering Context-Aware Object-Oriented Systems and Environments*, Seattle, USA, 2002.
11. Ranganathan, A., Campbell, R. A Middleware for Context-Aware Agents in Ubiquitous Computing Environments, *in ACM/IFIP/USENIX International Middleware Conference*, Rio de Janeiro, Brazil, 2003.
12. Ranganathan A. and Campbell R. H. An infrastructure for context-awareness based on first order logic, *Personal and Ubiquitous Computing*, 7(6):353–364, 2003.
13. Wang X. H. et al. Ontology Based Context Modeling and Reasoning using OWL, *Workshop on Context Modeling and Reasoning at IEEE International Conference on Pervasive Computing and Communication (PerCom'04)*, Orlando, Florida, March 14, 2004.
14. Disappearing Computer initiative website http://www.disappearing-computer.net/
15. extrovert-Gadgets project website http://www.extrovert-gadgets.net
16. ISTAG website http://www.cordis.lu/ist/istag.htm
17. Semantic Web in UbiComp Special Interest Group. http://pervasive.semanticweb.org