A Formal Framework for Modeling Context-Aware Behavior in Ubiquitous Computing

Isabel Cafezeiro[1], José Viterbo[2], Alexandre Rademaker[2], Edward Hermann Haeusler[2], and Markus Endler[2]

[1] Departamento de Ciência da Computação
Universidade Federal Fluminense
Rua Passo da Patria, 156 - Bloco E - 3 andar, Boa Viagem
24210-240 – Niterói – Brasil
[2] Departamento de Informática
Pontifícia Universidade Católica do Rio de Janeiro
Rua Marquês de São Vicente 225, Gávea
22453-900 – Rio de Janeiro – Brasil

Abstract. A formal framework to contextualize ontologies, proposed in [3], provides several ways of composing ontologies, contexts or both. The proposed algebra can be used to model applications in which the meaning of an entity depends on environment constraints or where dynamic changes in the environment have to be considered. In this article we use this algebra to formalize the problem of interpreting context information in ubiquitous systems, based on a concrete scenario. The main goal is to verify, on one hand, how the formal approach can contribute with a better understanding of the fundamental concepts of ubiquitous computing and, on the other hand, if this formal framework is flexible and rich enough to adequately express specific characteristics of the concrete application domain and scenario.

1 Introduction

In the last years, Ubiquitous Computing has been the focus of much research, most of which in topics related to system's development. Despite that, until now, very few works can be found on formal models for this area, where the main challenge is to precisely model — and reason about — the interactions between a system and its environment, and the fact that this environment can change in unpredictable ways.

In particular, context-awareness, i.e. the ability of applications to detect changes in their environment and to adapt their behavior accordingly, has soon become the paramount programming paradigm for such systems. As a consequence, more recently, many researchers have attempted to define, classify or model the notions of context and context-awareness. Nevertheless, most of these definitions are informal and thus lack a solid, mathematical foundation. Therefore, according to several authors [1,5], there is still demand for a comprehensive

T. Margaria and B. Steffen (Eds.): ISoLA 2008, CCIS 17, pp. 519–533, 2008.

formal framework for understanding and working with ubiquitous and context-aware computing.

For such a framework to be useful, however, it should ideally: (a) closely reflect the intrinsic characteristics of ubiquitous systems, clearly describing their relevant issues, and (b) provide means of describing (and solving) concrete application problems by allowing a suitable interpretation adaptated from the results of the underlying theory.

In [3] we have proposed a formal framework to contextualize ontologies, providing several ways of composing ontologies, contexts or both. This algebra is suitable for modeling applications in which the meaning of an entity depends on environment constraints or where dynamic changes in the environment should be considered. It emphasizes the relationships of contexts with entities — considering that contexts are essential to assign meaning to entities — and supports new forms of representing context for applications that consider dynamic changes of the environment. In this article we use this algebra to formalize the problem of interpreting context information in ubiquitous computing systems.

Through this experiment we intend to show not only how the formal approach may contribute with a precise understanding of the concepts and fundamental problems of a specific application domain, but also, how a concrete application domain can be used to assess the flexibility and expressiveness of a formal language. We believe that this is essential for reducing the gap between the theoretical framework and its possible applications, and for validating its mechanisms in a concrete and complex application domain.

The formal framework considered here is founded in Category Theory [4,8]. Along this article, though, we avoided the use of the categorical terminology, preseting concepts in an informal way. In [2] and [3] the reader can find the formalization of the algebra and associations between categorical concepts and the ontology terminology.

This article is organized as follows. In Section 2 we describe other efforts to formalize ubiquitous and context-aware systems. In Section 3 we discuss the algebra of contextualized ontologies. In Section 4 we describe ubiquitous environments and present a specific scenario. In Section 5 we apply the algebra to formalize the ontologies discussed in this scenario. Finally, in Section 6 we present our conclusions.

2 Related Work

In the last few years, some research has been undertaken towards formalizing ubiquitous and context-aware computing. Roman et al [6,10] have proposed Context UNITY, a dialect of Mobile UNITY with constructs that allow the reasoning about the manipulation of context, as well as the interaction of systems with the context. Their goal was to specify applications that use flexible mechanisms for defining individual contexts, which are transparently maintained as the environment evolves. In their approach, context is defined from the perspective of each component and hence, not every component sees the same context. On one

hand, the variable-assignment-notation of Context UNITY — used to express context definitions and resolutions — is quite expressive, but on the other hand, it is quite complex to be used for larger and more sophisticated context-aware applications.

A completely different approach is pursued in [15], where the authors present an extension of the classical action system formalism with the notion of context. In their formalism it is possible to prove some system properties using standard action system proof techniques. What the authors call a *Context-Aware Action System* is built from a parallel or prioritized composition of simpler action system components, where the context dependency of each component is expressed as a collection of relations that constrain when the computation of the component can take place, and how its internal actions are affected by the context variables. By applying their formalism in two small — but concrete — context-aware examples (i.e. a reminding and a messaging service), they show that their formal framework can be helpful to modularly describe systems and infer some simple context-aware properties.

Birkedal et al [1] investigated the modeling of context-aware systems using Bigraphical Reactive System (BRS) Models (i.e. graphical models of mobile computation that emphasize both locality and connectivity and a set of reaction rules that rewrite bigraphs to bigraphs. According to the authors, the main goals of the theory of Bigraphical Reactive Systems are to model ubiquitous systems on one hand, and to be a meta-theory encompassing existing calculi for concurrency and mobility (such as CCS, π-calculus) on the other hand. Hence, this work tried to push forward the first goal. Interestingly, however, the main finding was that naively modeling context-aware systems as BSRs is very complex and awkward. Instead, they have proposed a model named *Plato-graphical* model, where the system's perceived context and the actual context are represented as distinct but overlapping BRSs. Using their formalism they described a simple location-aware printing service, and concluded that the resulting model was very well suited for modeling the location-aware systems.

Our proposal differs from Context UNITY, but can serve as a model for the later. In fact, UNITY has a precise Category Theoretic based semantics that fits quite well in our approach and allows a better comparison between the approaches. While Context UNITY puts the context information inside the specification (by means of the *context* section), our approach maps the specification into each of the environments listed in this very context section. Therefore, one can say that our approach is structurally and semantically richer than Context UNITY. On the other hand, the approach using Context-Aware Action System is more related to the way current context-aware systems are designed, i.e. in which some of their actions are triggered, or inhibited, by specific conditions of the environment. Hence, this approach is much more related to an operational view of systems (e.g. syncronous composition on sequential processes) than ours, but which can be adequately interpreted as categorial co-limits induced by the actions alphabet. Finally, the approach based on Bigraphical Reactive Systems, as long as it serves as a model for process algebras in a true-concurrency style of

semantics, can be regarded as being at a similar level of abstraction as ours. However, since the bigraphical representation model seems to lack compositionality, we believe that our approach is more adequate for modeling complex systems.

3 The Algebra of Contextualized Ontologies

The algebra of contextualized ontologies is designed for applications in which additional information is required in order to describe an entity. This information, that we call *context*, may be some kind of meta-data or any information related to — but not particular to — that entity. This is the case of ubiquitous computing applications [14]. Under this paradigm, information concerning either physical or computational environment is a relevant part of the application. Besides, the overall information available for an application — i.e. the context where it is imersed — constantly suffers dynamic changes.

This algebra is based on two basic features: *(i)* a uniform representation of entities and context and *(ii)* the emphasis on the relationship. Concerning *(i)*, we use ontologies for representing both entities and contexts. This enhances the flexibility of the framework avoiding to determine *a priori* the role of an ontology: an ontology may represent an entity, a context or even both an entity and a context. Concerning *(ii)*, the framework puts the focus on the *relationship* among the components of a systems and not on the components themselves. In this way, the internal constitution of an entity is hidden, and descriptions are built in a modular and reusable way. The benefits of emphasising relationships are similar to those well known in systems constructions since the 70's: *Every module (...) is characterized by its knowledge of a design decision which it hides from all others. Its interface or definition was chosen to reveal as little as possible about its inner workings* [9]. The combination of *(i)* and *(ii)* makes possible the reuse of descriptions in a wide sense. Also, as it is the relationship that determines, at any time, the role of an ontology as entity or context the meaning of the subject being described is given by a net of relationships, what enables more accurate descriptions.

3.1 Contextualized Ontologies

By ontology we refer to a structure composed by concepts organized in a taxonomy, relations that determine non-taxonomical relationships, and logical axioms that set restrictions among relationships. The axioms are given in some expressive language whose model-theoretic semantics provides meaning.

Contextualized Ontologies are described as structures that persist a link between two ontologies. The source of the link is the entity and the target is the context. By *structure preserving* we mean that the context respects the hierarchical structure and the ontological relations of the entity. In other words, the entity is coherent with respect to its context. Formally, this means that if an ontology O has a relation $f(c_1, c_2)$ where c_1, c_2 are concepts of the ontology. Then a link $F : O \to O'$ from O to a context O' is such that $F(f(c_1, c_2)) = F(f)[F(c_1), F(c_2)]$.

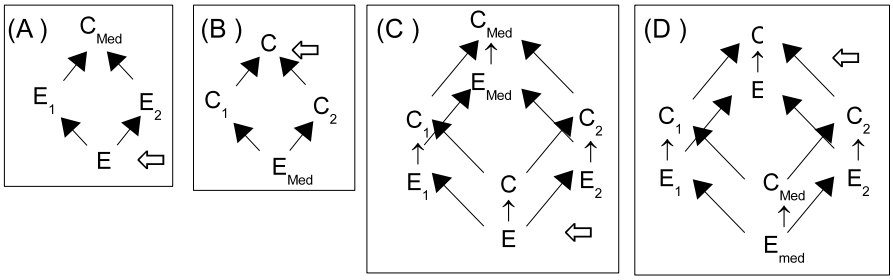

Fig. 1. (A) Entity Integration. (B) Context Integration. (C) Relative Intersection. (D) Collapsed Union.

In order to avoid violating internal constitution of entities, few constraints must be stated about links: *(i)* there is an identity link for any entity or context, that maps the entity/context to itself. Thus the entity may be viewed as a (non-informative) context of itself; *(ii)* an entity is called *domain* of a link, while a context is called *codomain* of a link; *(iii)* links can be composed in an associative way if the codomain of the first is the domain of the second. The notation of a triple (entity, link, context), also represented by $e \rightarrow c$, will be used any time we want to identify the ontologies that act as entity or context in a contextualized ontology. We will use the symbol "$\circ$" to denote the associative composition of contextualized ontologies.

In the sequel we present modular constructs that can be applied to contextualized entities, in order to coherently combine entities, contexts or both. We divide the operations in three classes: Entity Integration, Context Integration and Combined Integration. We use the term "component" to refer to concepts or relations of ontologies[1].

Entity Integration. (Fig. 1-A) Operations in this class have the purpose of integrating entities (E_1 and E_2) that share the same context:$E_1 \rightarrow C_{Med} \leftarrow E_2$. As entities are coherent with respect to their context, the integration has the context as mediator. The result is a new entity (E) contextualized by the original ones (and by transitivity, by the original context C_{Med}). The entity integration performs the semantic intersection of the entities under the mediation of the context, that is, the new entity will embody all, and nothing more than, information of the original entities that is mapped in the same component of the context.

Context Integration. (Figure 1-B) These operations consider situations where a single entity E_{Med} has more than one context ($C1$ and $C2$) : $C_1 \leftarrow E_{Med} \rightarrow C_2$. The context integration produces a new context $C_1 \rightarrow C \leftarrow C_2$ that combines

[1] The reader aware of Category Theory will note that in the following discussion one is considering a category of Ontologies $\mathcal{O}$ and the operations just described correspond to limits and colimits taken in $\mathcal{O}$ itself and $\mathcal{O}^{\rightarrow}$ respectively.

information of the original context preserving the coherence with the entity. This operation can be used in situations where a single entity can be viewed in many ways, according to the considered context. The integration performs the amalgamated union of contexts, collapsing components that are images of the same component in the original entity.

Combined Integration. This class of operations embodies two subclasses: relative intersection and collapsed union. They consider the need to integrate the contextualize ontology as a whole, whithout making distintion between entity or context.

Relative Intersection. (Figure 1-C) Is the intersection of two contextualized ontologies mediated by a third contextualized ontology. It produces a new contextualized ontology having just the components of the originals that are mapped in the mediator.

Collapsed Union. (Figure 1-D) Is the amalgamated union of two contextualized ontologies mediated by a third contextualized ontology. It produces a new contextualized ontology having all components of the original but collapsing those components of the original that are image of the same component of the mediator.

4 Ubiquitous Computing

Ubiquitous computing is a particularly interesting and challenging domain for applying the formal algebra described in Sect. 3. In the vision of ubiquitous computing, computer systems will seamlessly be incorporated into our everyday lives, providing services and information anytime and anywhere [14]. Compared to traditional distributed systems, ubiquitous systems feature increased dynamism and heterogeneity. The underlying ubiquitous computing infrastructures are more complex and bring into the foreground issues such as user mobility, device disconnections, join and leave of devices, heterogeneous networks, as well as the need to integrate the physical environment with the computing infrastructure [7].

As a fundamental requirement, ubiquitous applications must be capable of responding to dynamic changes in their environments with minimal human interference. Users should be able to take full advantage of the local capabilities within a given environment and be able to seamlessly roam between several environments, despite variations of the computing and communication resources' availability (e.g. available wireless bandwidth, residual energy, location-specific services, etc). Hence, ubiquitous computing systems strongly rely on context data, which is used to trigger adaptations at different levels, such as at communication protocols, middleware services, or the user interface.

In ubiquitous systems, ontology has been widely adopted for representing context information. The use of ontologies has not only the advantage of enabling

the reuse and sharing of common knowledge among several applications [11], but also of allowing the use of logic reasoning mechanisms to deduce high-level contextual information [13]. In the following subsection we describe a simple scenario, which illustrates the use of context in a ubiquitous environment, and highlights some concepts such as location-specific context, reasoning, heterogeneous contexts and semantic mediation. The numbers in italics between brackets are used to identify situations that will be further referred in Sect. 5.

4.1 Scenario

We consider two universities in Brazil, for instance, PUC-Rio and UFF, which are collaborating in some research projects, e.g. the UbiForm Project. Silva is a professor and researcher who works at the CS Department of PUC-Rio, and is also participating in the UbiForm Project. Silva carries with him his smart phone, which host some context-aware applications that respond to different situations, according to his preferences and to environment conditions.

When he arrives at PUC-Rio, an Ambient Management Service (AMS) registers his smart phone (SMP_{Silva}) and detects that it belongs to him. The system verifies that Silva works there as a professor and sets his workspace *(1)*. This service also informs other members of Silva's team about Silva's arrival *(2)*. A Personal Agenda application running on SMP_{Silva} contacts the context infrastructure with a request to be notified about the beginning of each event involving the whole project team, based on the project schedule and the location *(3)*. Another application on SMP_{Silva}, a Configuration Management Service (CMS), requests to be notified whenever Silva is in a room in which an activity (e.g. a technical presentation, a brainstorm session) has started, so that it may set the smart phone to blocked mode, and as soon as the activity ends, switch it back to the ring mode. But if Silva's wife sends him a message during the meeting, the phone should vibrate, so that he can discreetly check the message's subject *(4)*.

From this example, we may see that the ubiquitous services described above rely on a wide variety of context information to trigger their actions. While the Ambient Management Service and the Personal Agenda must be aware of the context information that describes Silva's role and location in the organization, the Configuration Management Service also takes into consideration Silva's personal preferences. Thus, to be able to apply the rule described, we notice that the context that fully describes the user Silva comprises not only the context that describes his role at PUC-Rio (location of Silva and his device in the organization), or in the UbiForm Project (schedule of activitues), but also the context that describes Silva's personal preferences and features (the one calling is Silva's wife). When Silva is at home or somewhere else — e.g. at an Airport *(5)* —, the Configuration Management Service will be imersed in an different overall context. In such cases, formalization may help to describe and understand how different contexts form a specific combined view.

Now supposing that Silva is visiting UFF with several other researchers and, as usually, he carries with him his smart phone running the same context-aware

services. Their purpose is to have joint workshops about the collaboration project. When Silva arrives at UFF, the Wi-Fi and GPS enabled SMP_{Silva} connects to the network, and using the current GPS data, queries a location service to find out that its owner (Silva) is at UFF *(6)*. It then determines that this university is a partner institution of PUC-Rio; obtains the IP address of the AMS at UFF and registers with it, indicating the user's identity and preferences. The Ambient Management Service registers SMP_{Silva} and identifies that the device belongs to Silva, a visiting professor from PUC-Rio. The system verifies that Silva is involved with the collaboration project and sets a workspace for him *(7)*. Notice that when the Personal Agenda and the Configuration Management Service interact with the Ambient's local context provider at UFF, although Silva is identified as a visitor at that institution, he can still be perceived as a professor from PUC-Rio. Hence, supposing that only professors can have access to printers at UFF, when setting Silva's workspace, AMS will recognize this access permission and configure the printer setup utility at his operating system to use the locally available printers *(8)*. In addition to this, suppose that AMS would make available to Professor Silva the publications of UFF which are related to his production. For this, AMS should also be aware of Professor Silva's production, i.e. list of publications. Once more, we identify that one of the main requirements of ubiquitous systems is the ability to adapt services/behaviors to the current context view. Again, formalization may be useful to describe a relation between different contexts in the form of a resulting aligned view.

5 Formalizing the Application

5.1 High Level Diagrams

In this section, we refer to the numbers that appear in Sect. 4.1 to draw high level diagrams of the situations described in the scenenario. Consider that *Silva*, *PUC*, *UFF*, *UbiForm Project* and *Airport* are ontologies that, describe, respectively, personal information about Professor Silva, PUC-Rio and UFF administrative organization, and information about the UbiForm Project and a given Airport. These ontologies are not contextualized. Their contexts will appear as we proceed in the construction of the formal model.

We start at *(1)*, when the Ambient Management Service (AMS) registers Professor Silva's smart phone. This process concerns the integration of the ontologies *Silva* and *PUC* with respect to the smart phone of Professor Silva. We construct a very simple ontology: SMP_{Silva} to be contextualized in *Silva* and *PUC*. This means that concepts and relations of SMP_{Silva} will be linked into correspondents of *Silva* and *PUC*, respecting the structure of the ontologies. SMP_{Silva} will act as mediator of *Silva* and *PUC* in a context integration $Silva \overset{AMS}{\longleftarrow} SMP_{Silva} \overset{AMS}{\longrightarrow} PUC$. The integration will result in a new ontology that we will name $SilvaAtPUC$.

It will embody all components of *Silva*, all components of *PUC*, and will have the images of concepts of the mediator SMP_{Silva} collapsed. Operating in

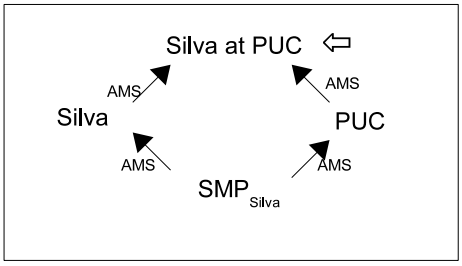

Fig. 2. Considering the ontologies of Professor Silva and PUC, AMS generates the ontology SilvaAtPUC

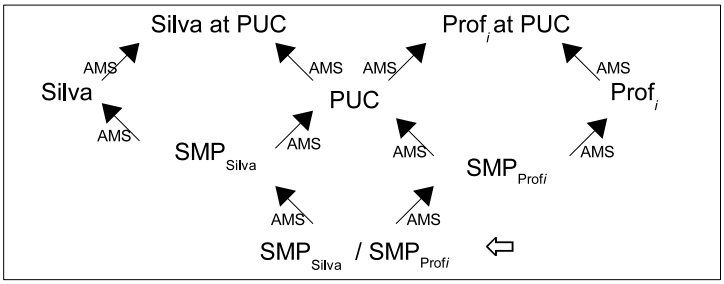

Fig. 3. AMS integrates SMP of Silva with SMP of professors

this integrated context ($SilvaAtPUC$), AMS will have enough information to identify the presence of Professor Silva at PUC (Fig. 2). A similar diagram can be considered for each member of the project that is present at the moment.

Then, in *(2)*, AMS informs other members of Silva's team about his arrival. Considering that, for any member $Prof_i$, a context integration $Prof_i \xleftarrow{AMS} SMP_{Prof_i} \xrightarrow{AMS} PUC$ has been generated, the entity integration of each SMP_{Prof_i} and SMP_{Silva} under the context of PUC will make the connection among the smart phones of the i professors of PUC and the smart phone of Professor Silva (Fig. 3). The resulting entity will be composed by the smart phone of each professor.

In *(3)*, the Personal Agenda (PA) of Silva's smart phone contacts the UbiForm Project Agenda to be notified about scheduled activities. The entity integration $SMP_{Silva} \xrightarrow{PA} UbiFormProject \xleftarrow{PA} SMP_{Prof_i}$ embodies the synchronization of the professors' agendas with respect to the UbiForm Project agenda. In the resulting ontology the Personal Agenda can process information about events in which all professors i and Silva take part (Fig. 4).

In *(4)* the Configuration Management Service (CMS) (running on SMP_{Silva}) requests the UbiForm Project Agenda to be notified when any activity is about to start. AMS is aware of the location of Professor Silva at PUC, and hence of his presence in a room where a project activity is taking place. It also considers

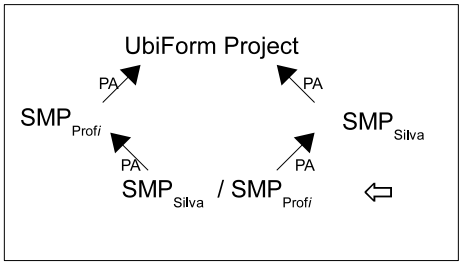

Fig. 4. Personal Agenda of Silva's smart phone contacts the UbiForm Project Agenda to be notified about scheduled activities, and to be synchronized with the other professor's agends

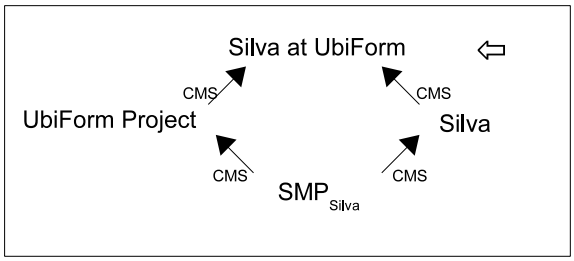

Fig. 5. CMS considers personal information about Silva and his physical position at the UbiForm

Silva's personal information in order to properly configure his phone alarm. This phone configuration could be represented by a rule — involving concepts associated to different contexts — that would trigger an adaptation for the CMS application:

$$\textbf{Device}(SMP_{Silva}) \wedge \textbf{isLocatedIn(?d,?r)} \wedge \textbf{inActivity(?r)} \wedge \textbf{PersonCalling(?p)} \wedge \textbf{isWife(?p,“Silva”)} \Rightarrow \textbf{setVibrate}(SMP_{Silva})$$

A context integration $UbiFormProject \overset{CMS}{\longleftarrow} SMP_{Silva} \overset{CMS}{\longrightarrow} Silva$ results in a context $SilvaAtUbiForm$ which combines personal information about Silva and the present UbiForm activity in which he is involved (Fig. 5). Similar situation occurs when Silva is somewhere else, e.g. as at the airport *(5)*. The context integration $Airport \overset{CMS}{\longleftarrow} SMP_{Silva} \overset{CMS}{\longrightarrow} Silva$ results in the context $SilvaAtAirport$ wherein the CMS can configure his phone alarm according to his contextual preferences.

Later, Professor Silva is visiting UFF *(6)*, where he is registered as a visitor researcher. Within the context $SilvaAtUFF$ that results from integration $Silva \overset{AMS}{\longleftarrow} SMP_{Silva} \overset{AMS}{\longrightarrow} UFF$, AMS can properly set the professor's workspace. But some of Silva's permissions for the use of resources come from the fact that he is a Professor at PUC *(7)*. Thus, information about Silva's

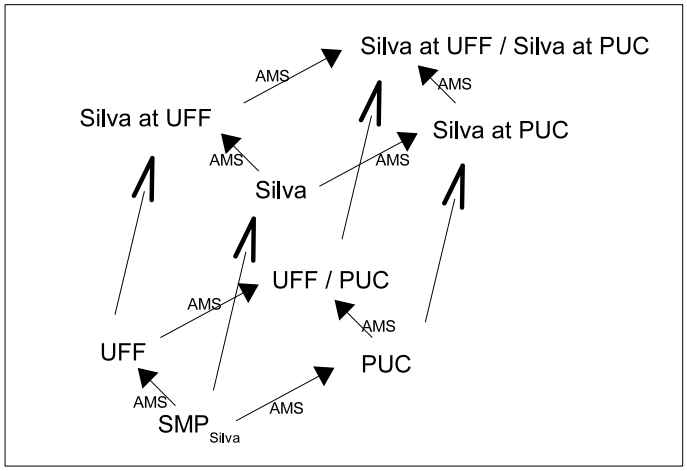

Fig. 6. Each face of the cube shows a context integration. The cube can also be considered as the collapsed union of the contextualized entities $UFF \rightarrow SilvaAtUFF$, $PUC \rightarrow SilvaAtPUC$ mediated by $SMP_{Silva} \rightarrow Silva$.

status at PUC must also be taken into account for setting access permissions properly. The context integration $UFF \overset{AMS}{\longleftarrow} SMP_{Silva} \overset{AMS}{\longrightarrow} PUC$ generates a context where AMS can find information about Silva as a PUC professor and as a UFF visitor researcher in the joint project UFF/PUC (base square of Fig. 6). The context integration $SilvaAtUFF \overset{AMS}{\longleftarrow} Silva \overset{AMS}{\longrightarrow} SilvaAtPUC$ generates a context where AMS can find not only information about Silva as a PUC professor or as a UFF visitor researcher, but also personal information about Silva (top square of Fig. 6). Note that Fig. 6 also pictures a combined integration: the collapsed union of the contextualized entities $UFF \rightarrow SilvaAtUFF$, $PUC \rightarrow SilvaAtPUC$ mediated by $SMP_{Silva} \rightarrow Silva$.

5.2 A Zoom into Ontologies and Morphisms

Since a detailed description of the whole scenario would exceed the space limitation of this paper, we selected only two diagrams of the previous subsection to illustrate how this framework provides the required information to adapt services or behaviors according to the context changes. First, we consider a situation in which information coming one context enables decisions about an entity in a different context. For instance, *(8)*, Professor Silva is allowed to use the printer at UFF as a consequence of the fact that, at PUC, he is a professor. AMS also makes available to Professor Silva the publications of UFF which are related to his production. The permission to print could be represented as a rule that would set an access permission in a ubiquitous regulation service, such as in [12]:

$$\textbf{Person(?p)} \wedge \textbf{worksAt(?p,“PUC-Rio”)} \wedge \textbf{playsRole(?p,“Professor”)} \Rightarrow \textbf{hasAccess(?p,“Printer”)}$$

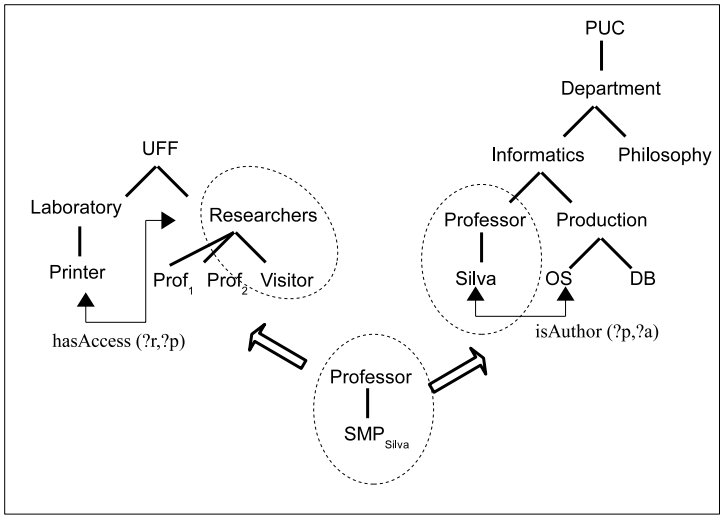

Fig. 7. Alignment of UFF and PUC under the mediation of SMP_{Silva}. The mediator captures the fact that Silva is a professor and properly map this information in the ontology of UFF.

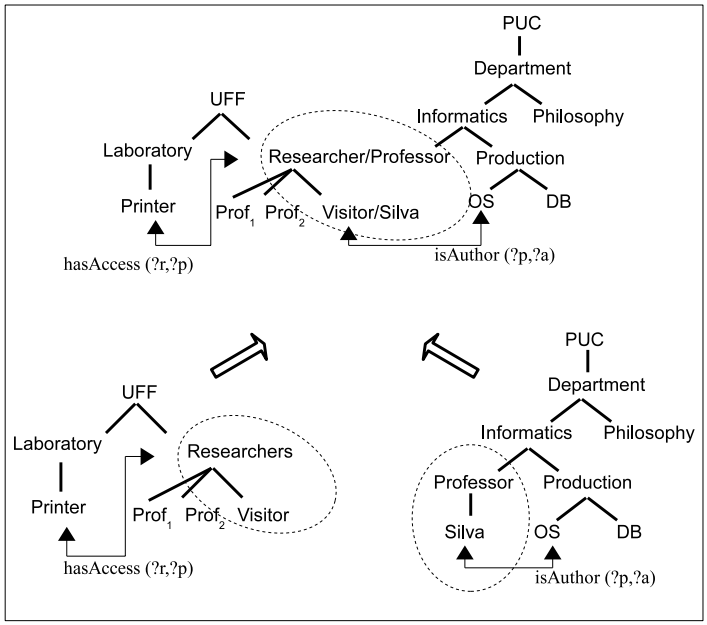

Fig. 8. The context integration of the alignment of figure 7: the relation *hasAcces(Researcher, Printer)* holds for Professor Silva and Printer and information about Professor Silva's production is avaiable

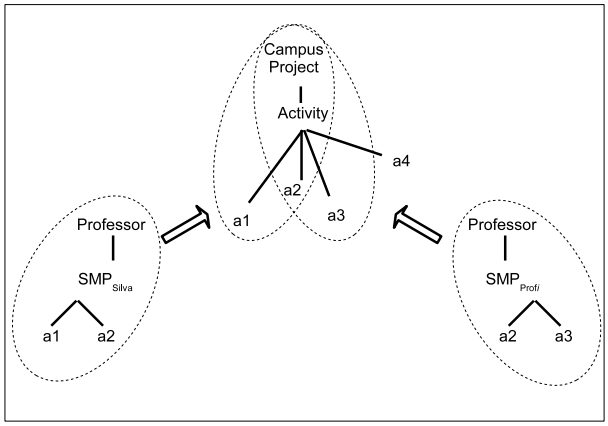

Fig. 9. The alignment of SMP of Professor i and SMP of Professor Silva with respect to the agenda of the UbiForm Project

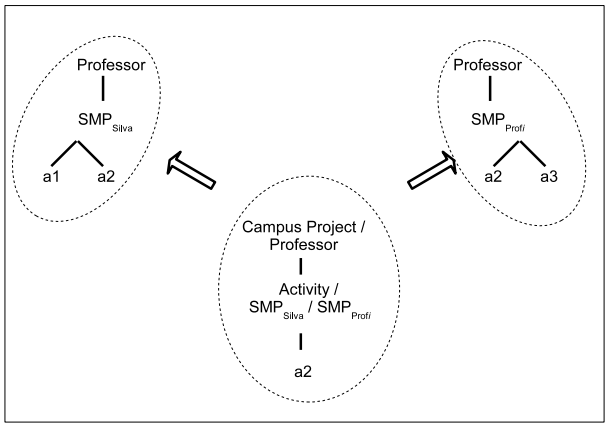

Fig. 10. Integration of agendas: Silva and Professor i will be present at Event 2

Considering the base square of Fig. 6, the mediator SMP_{Silva} of the context integration $UFF \overset{AMS}{\longleftarrow} SMP_{Silva} \overset{AMS}{\longrightarrow} PUC$ must capture the fact that Silva is a professor and properly map this information into the ontology of UFF. Figure 7 depicts the ontology for UFF and PUC and shows this alignment. Note that, as the concept *Professor* at *PUC* is related to *Researcher* at *UFF*, the relation *hasAccess(?p,?d)* will hold for Professor Silva and Printer in the resulting context (in Fig. 8). Also, note that, in this resulting context information about Professor Silva's production is avaiable to be used by AMS. Secondly, we show how the integration can filter information in order to affect just a selected set of entities. We consider the situation *(3)*, where the Personal Agenda of Silva's smart phone contacts the UbiForm Project Agenda to be notified about events.

Diagram of Fig. 4 pictures this situation, showing the integration of SMP of Professor i and SMP of Professor Silva under the context of the UbiForm Project. Figure 9 shows the alignment of SMP of Professor i and SMP of Professor Silva with respect to the context of the UbiForm Project. Figure 10 shows the resulting entity, in which appears only the events that both take part.

6 Conclusions

In this article, we used an algebra of contextualized ontologies to formalize the problem of interpreting context information in ubiquitous systems based on a concrete scenario. The main goal was to verify not only the contributions of this formal approach for better understanding of the fundamental concepts of ubiquitous computing, but also the adequability, expressiveness and flexibility of this approach to express specific characteristics of the concrete application domain and scenario.

Before using a formal model, method or language for a specific problem domain, it is worth thinking about the expected benefits versus the required efforts of this endeavor. In fact, formalization usually helps to develop a better understanding of the problem domain and its scope, as well as clearly define the major concepts involved. Ontologies strongly contributes in this sense, enabling modular and reusable taxonomical descriptions and enhancing the expressive power through the use of logic reasoning mechanisms. The algebra of contextualized ontologies enforce these benefits, having modularity and reuse as fundamental requirements, over which the operations to compose and decompose ontologies are defined. In the algebra, the alignment of ontologies is naturally supported as initial step of integration. As a result, the use of the algebra becomes very close to the usual way of handling ontologies.

One should also be aware of the limitations and potential risks of applying formal methods to a concrete problem. When we use a formal model for any subject we always abstract from some issues or entities which apparently seem less relevant. In real systems these issues might well have a significant impact on the real system's behavior, and should ideally be accounted for. Hence, whenever we develop a formal model of a system, there is always a trade-off between the model's degree of realism, its complexity and its underlying set of applicable basic results. The possibility of adopting levels of abstraction, however, contributes to reduce the gap between the formal approach and the real system. High level diagrams considering just entities and contexts give an abstract view of formalizations. Ontologies and mappings come later, in refinement steps introducing more details gradually.

References

1. Birkedal, L., Debois, S., Elsborg, E., Hildebrandt, T., Niss, H.: Bigraphical Models of Context-aware Systems. Technical Report TR-2005-74, The IT University of Copenhagen (November 2005)
2. Cafezeiro, I., Haeusler, E.H.: Semantic interoperability via category theory. In: Conferences in Research and Practice in Information Technology, vol. 83 (2006)

3. Cafezeiro, I., Rademaker, A., Haeusler, E.H.: Ontology and Context. In: Proceedings of CoMoREA 2008, pp. 53–62 (2008)
4. Goldblatt, R.: Topoi: The Categorical Analysis of Logicser. Ser Studies in Logic and the Foundations of Mathematics. North Holland, Amsterdam (1979)
5. Henricksen, K., Indulska, J.: Developing context-aware pervasive computing applications: Models and approach. Pervasive and Mobile Computing 2(1), 37–64 (2006)
6. Julien, C., Payton, J., Roman, G.-C.: Reasoning About Context-Awareness in the Presence of Mobility. In: Proc. of the 2nd Int. Workshop on Foundations of Coordination Languages and Software Architectures (FOCLASA 2003), Marseille, France (September 2003)
7. Kindberg, T., Fox, A.: System software for ubiquitous computing. Pervasive Computing Magazine (2002)
8. MacLane, S.: Categories for the Working Matematician. Springer, Berlin (1997)
9. Parnas, D.: On the criteria to be used in decomposing systems into modules. Communications of the ACM (December 1972)
10. Roman, G.-C., Julien, C., Payton, J.: A Formal Treatment of Context Awareness. In: Wermelinger, M., Margaria-Steffen, T. (eds.) FASE 2004. LNCS, vol. 2984, pp. 12–36. Springer, Heidelberg (2004)
11. Shehzad, A., Ngo, H.Q., Pham, K.A., Lee, S.Y.: Formal modeling in context aware systems. In: Proceedings of the First International Workshop on Modeling and Retrieval of Context (September 2004)
12. Viterbo, J., Endler, M., Briot, J.-P.: Ubiquitous service regulation based on dynamic rules. In: Proceedings of the 13th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS 2008), Belfast, pp. 175–182. IEEE Computer Society Press, Los Alamitos (2008)
13. Wang, X.H., Zhang, D.Q., Gu, T., Pung, H.K.: Ontology based context modeling and reasoning using OWL. In: Proc. of 2nd IEEE Conf. Pervasive Computing and Communications (PerCom 2004), Workshop on Context Modeling and Reasoning, Orlando, Florida, March 2004, pp. 18–22. IEEE Computer Society Press, Los Alamitos (2004)
14. Weiser, M.: The computer for the twenty-first century. Scientific American 265(3), 94–104 (1991)
15. Yan, L., Sere, K.: A Formalism for Context-Aware Mobile Computing. In: Proceedings of the IEEE 3rd International Workshop on Algorithms, Models and Tools for Parallel Computing on Heterogeneous Networks (HeteroPar), Cork, Ireland (July 2004)