Designing Ubiquitous Applications: Proposal of a Specification Environment

Isabel Cafezeiro
Departamento de Ciência da Computação
Universidade Federal Fluminense
Rua Passo da Patria, 156 - Bloco E - 3º andar,
Boa Viagem, 24210-240, Niterói, Brasil
isabel@dcc.ic.uff.br

José Viterbo, Alexandre Rademaker,
Edward Hermann Haeusler,
Markus Endler
Departamento de Informática
Pontifícia Universidade Católica
do Rio de Janeiro
Rua Marquês de São Vicente 225, Gávea,
22453-900, Rio de Janeiro, Brasil
{viterbo, arademaker, hermann,
endler}@inf.puc-rio.br

ABSTRACT

In ubiquitous applications, where the meaning of an entity, such as a user or service, depends on environment-specific constraints and dynamic changes in the environment have to be considered in all stages of development, the separation between the system's behaviour and its context representation (a.k.a. context model) is essential for facilitating the development of such inherently complex systems. At the same time, because of its well-known benefits, a formal specifications should be considered not only for describing the system's behaviour, but also the corresponding context model. Considering this, we propose in this paper an environment to support context modelling through formal specification. For this sake, we adopt the algebra of contextualized entities proposed in [2, 3] and define levels of abstractions over its diagrams, enabling a stepwise construction of modular specifications. The overall goal is to reduce the gap between the formal description of an ubiquitous application and its implementation.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Distributed Systems; F.3.1 [**Logics and Meanings of Programs**]: Specifying and Verifying and Reasoning about Programs—*Specification techniques*

General Terms

Design, Theory

Keywords

Context-awareness, Formal Specification, Ubiquitous Systems

MUCS2009, June 15, 2009, Barcelona, Spain.

1. INTRODUCTION

The adoption of a formal method or language usually helps to reach a better understanding of the problem domain and contributes to clarify the concepts involved. Moreover, as a formalization is embedded in a wider theoretical framework, with its theorems and results, by presenting an application problem in terms of a more abstract theory it is sometimes possible to adapt theoretic results to the application problem, yielding interesting, and previously unforeseen, results, such as about the inherent complexity of the problem at hand. Moreover, the effort to implement the automatic generation of programs and validating properties of programs are usually minimized within the theoretical framework.

On the other hand, when we use a formal model we abstract from some issues or entities which apparently seem less relevant. However, in real systems, these issues might well have a significant impact on the real system's behavior, and should ideally be accounted for. Hence there is always a trade-off between the model's degree of realism, its complexity and the set of applicable results derived from the model.

Considering the additional complexity of context-awareness, where the meaning of an entity, such as a user or service, depends on environment-specific constraints and dynamic changes in the environment have to be taken into account in all stages of development, we stress the importance of considering a system (i.e. its behaviour) and the context model as separate domains. This separation tends to generate appropriate representations for the information of context, which therefore allows the design of simpler algorithms to specify the behaviour of the system.

Thus, a formal treatment to context-awareness may contribute with the field by providing a better understanding of concepts and mechanisms, giving insights to implementation decisions, making possible the anticipation of formal verifications to development time, suggesting new mechanisms of software construction focusing the obtainment of trustworthy systems, and finally, offering the possibility of adapting results of the theoretical framework to the applied problem.

In this paper we adopt a formal algebra and propose an environment for specifying ubiquitous systems with the goal of reducing the gap between formalism and implementation.

This focus is present both in the algebra and in the design of the environment:

The algebra ([2, 3, 1]) hides the theoretical framework (Category Theory) under a suggestive terminology, takes *contextualization* as a basic notion and proposes a small set of useful operations to compose and decompose *contextualized entities*. It also employs a homogeneous representation of entities and contexts, and by emphasising their relationships, yields a simple, modular and uniform formalization.

The environment takes advantage of the fact that the algebra has a diagrammatic representation and split its diagrams in three levels of abstraction: diagram of devices and applications, diagram of entities and diagram of ontologies. These different levels are integrated and can be modified at any time. Thus, small reminders, tips or annotations that can be associated to elements of the diagrams can be, through transformations, propagated and refined according to the level of abstraction of each diagram.

1.1 Related Works

In [6], it is presented an approach where context is determined by a network of agents (similar to our entities). As in our approach, the net avoids restricting the context to data/information that an entity can directly sense and amplifies the notion of context to the extension of the net. In [6], *views* are used to control the information available to the agent. Every view is defined relative to an agent (reference agent) and with respect to its needs for resources from and knowledge about its environment. This role is similar of the *links* that connects entity and context in our approach. However in [6] agents and contexts do not have a uniform representation what causes a loss of flexibility.

CommUnity [8] is also based on Category Theory and emphasises the separation between application logic and context modelling. As it evolved from a previous work on distribution and mobility in software architectures, it adopt concepts of communication by channels and location variables, in what, strongly differs from the approach presented here.

1.2 Previous Works

The approach we present in this paper is result of a research project that evolved in the following way: In [1] we investigated benefits and limitations of using ontologies to achieve a high degree of semantic interoperability. We adopted Category Theory as explicit formal framework and formalized integration mechanisms over it. In [2] we proposed a formal framework to contextualize ontologies. Talking about entities and their contexts as contextualized ontologies, we presented an algebra to compose and decompose contextualized ontologies in several ways. Although serving as formal basis over which the whole algebra was constructed we presented algorithms and integration mechanisms without explicit use of Category Theory. In [3] we tested the applicability of formal framework described in [2] in a scenario of ubiquitous computing. Among its most relevant characteristics, ubiquitous systems must respond dynamically to changes in the environment, with litte or no human interference. Hence, they must be context-aware, which means that information about the context wherein they are supposed to operate is key to their functioning. Interoperability has been successfully modelled by Category Theory (as in formal specification of systems [4] and in software architecture [5]) and contexts are the essence of the algebra presented in [2]. This justifies the matching between the chosen formalism and the application paradigm.

In this paper we present the basis for the implementation of an environment of specification showing how the algebra can be split in three levels, and how information of these levels can be integrated.

This paper is organized as follows: In section 2 we present a brief description of the algebra. Section 3 is devoted to the presentation of the environment. For this, we adopt a concrete scenario (subsection 3.1) to be used as running example and describe the modules of the environment (subsection 3.2). We then describe how these modules are related (subsection 3.3). In section 4, we present an overview of the formalization of the scenario. Finally, in section 5 we conclude the paper.

2. BRIEF PRESENTATION OF THE ALGEBRA OF CONTEXTUALIZED ENTITIES

In this section we informally present the algebra of contextualized entities and its constructs. For the reader interested in the formal details of the algebra or in the algorithms for computing them, we point previous works [1, 2, 3].

The algebra of contextualized entities is based on three features: *(i)* a uniform representation of entities and context, *(ii)* the independence of representation and *(iii)* the semantics via net of relationships, where *(i)* provides interoperability and *(ii)* and *(iii)* stresses the emphasis on the relationship. The three features together provide flexibility, avoiding to determine *a priori* the role of a component as entity or context; give support to modularity and reuse, as a consequence of hiding the internal constitution of an entity; and enable more accurate descriptions, as the meaning of the subject being described is given by a net of relationships.

The algebra is defined over objects that we call *contextualized ontologies*. These objects are composed by three parts: an entity, a context and a link between them. Both entity and context are represented by ontologies and the link between them ensures that an entity is coherent with its context: structure and relations of the entity is reflected in the context; axioms of the entity, that are properly translated to the context language, hold in the context.

When referring to ubiquitous computing, we can think about the entity as being a computational device or a person, represented by a computational device. The context can be thought of as the environment where the entity operates, which can either be a physical environment or another computational device. The link is the immersion of the entity into its context – both entity and context are described as ontologies. Under this model, information concerning either physical or computational environment is treated as a relevant part of the application, deserving its own representation. This representation is self contained and not connected to the entities, what gives a high degree of mobility.

EXAMPLE 1. *Figure 1 shows a brief part of the scenario that will be detailed in section 4. The simple ontology at left represents a Professor Silva, who is recognized by his smart phone code (SMP). Silva is the entity. The link maps Silva in the context PUC-Rio, the institution where he works as professor.*

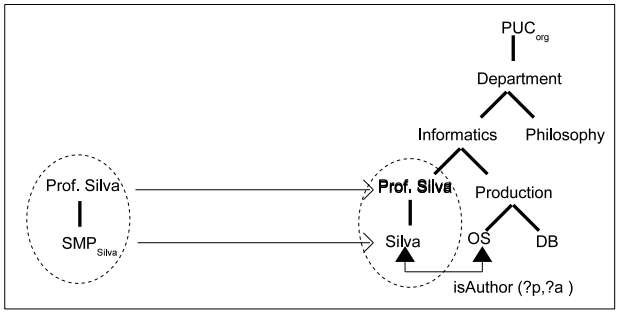

Figure 1: A contextualized entity: Silva at PUC-Rio.

2.1 Operations on Contextualized Ontologies

The algebra is formed by two sets of dual operations devoted to compose and decompose contextualized ontologies in several ways. For composing the operations are: *alignment*, *context integration*, *collapsed union*. For decomposing the operations are: *coalignment*, *entity integration* and *relative intersection*. The first set is devoted to produce the union of parts, where some components may be collapsed according to specific semantic identifications (semantic union). The second set is devoted to produce intersection, also considering specific semantic identification (semantic intersection). These identifications are specified by the links associations.

All the operations have the general purpose of manipulating information of ontologies in order to produce a coherent and concise body of information over which a particular device may operate. For this, the duals *align* and *coalign* play the role of stablishing binary relations between contexts or between entities. The duals *context integration* and *entity integration* compose/decompose context/entity respecting the entity/context. Finally, the duals *collapsed union* and *relative intersection* perform the semantic union / semantic intersection of contextualized entities as a whole.

Alignment. (Figure 2-A) "Is the task of establishing a collection of binary relations between the vocabularies of two ontologies. Since a binary relation can itself be decomposed into a pair of total functions from a common intermediate source, we may describe the alignment of two ontologies O_1 and O_2 by means of a pair of ontology mappings from an intermediate source ontology O."[7] Concerning contextualized ontologies, a situation where an entity has more than one context is an alignment: $C_1 \leftarrow E_{Med} \rightarrow C_2$. By defining

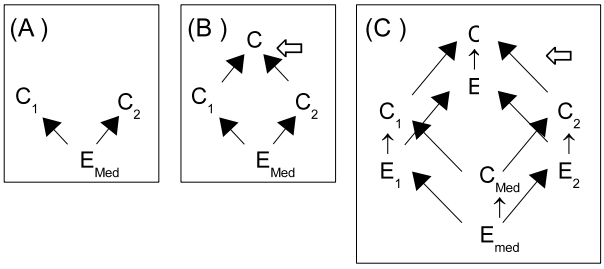

Figure 2: (A) Alignment. (B) Context Integration. (C) Collapsed Union.

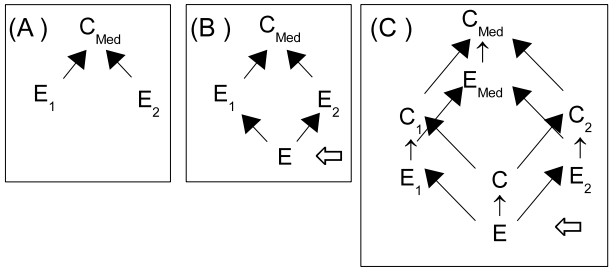

Figure 3: (A) Coalignment. (B) Entity Integration. (C) Relative Intersection.

a binary relation, the alignment makes possible the partial mapping between contexts. This feature makes possible to deal with situations where a concept of a context does not make sense in the other context. On the other hand, the entity must be totally mapped on both contexts: all concepts of the entity must be understood in both contexts.

Context Integration. (Figure 2-B) When 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, to which C_1 and C_2 are linked: $C_1 \rightarrow C \leftarrow C_2$. This new context combines information of C_1 and C_2 preserving the coherence with the entity E_{Med}. The integration performs the amalgamated union of contexts, collapsing components that are related by the alignment $C_1 \leftarrow E_{Med} \rightarrow C_2$. The result contains all information of the original contexts, but identifies parts related by the mediator entity. As the operation is guided by ontology links, structure, relations, and axioms of the mediator entity are preserved.

Collapsed Union. (Figure 2-C) Is the amalgamated union of two contextualized ontologies mediated by a third contextualized ontology. It is the combined composition of entities and contexts, where the ontology links ensures the preservation of structure, relations, and axioms of each ontology and coherence of each entity with respect to contexts. It produces a new contextualized ontology with all components of the original ones, but collapsing components that have the same source in the mediator.

Coalignment. (Figure 3-A) It is a mechanism of establishing a correspondence between vocabularies of two ontologies by the use of an intermediate target ontology. It configures a binary relation between the two ontologies, where related components are those that are mapped in the same component of the intermediate target. Concerning contextualized ontologies, a situation where two entities share the same context is a coalignment: $E_1 \rightarrow C_{Med} \leftarrow E_2$. By defining a binary relation, the coalignment makes possible the partial mapping between entities, what means that not all concepts of one entity make sense in the other entity. Both entities, however, must be totally mapped in the context.

Entity Integration. (Fig. 3-B) Is the integration of entities (E_1 and E_2) that share the same context: $E_1 \rightarrow C_{Med} \leftarrow E_2$. 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 are related by the coalignment $E_1 \rightarrow$

$C_{Med} \leftarrow E_2$. As the operation is guided by ontology links, structure, relations, and axioms are preserved.

Relative Intersection. (Figure 3-C) Is the intersection of two contextualized ontologies mediated by a third contextualized ontology. It is the combined intersection of entities and contexts, where the ontology links ensure the preservation of structure, relations, and axioms of each ontology and coherence of each entity with respect to its context. It produces a new contextualized ontology having just the components of the originals that are mapped in the mediator.

3. A SPECIFICATION ENVIRONMENT

In [3] we adopted a scenario to be formalized and to illustrate the operations of the algebra. In this paper we use the same scenario to illustrate the proposed environment, showing how the diagrams of the algebra can be constructed in different levels of abstractions. In subsection 3.1 we describe the scenario. In subsections 3.2 and 3.3 we describe the modules and their connections, using the scenario as running example.

3.1 The Scenario

We consider two universities in Rio de Janeiro, Brazil, 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. The Ubiform Project Agenda (UPA), a service of AMS, informs the members of UbiForm Project the about Silva's arrival. A Personal Agenda application running on SMP_{Silva} contacts UPA 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. 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.

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, 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 activities), but also the context that describes Silva's personal preferences and features (the vibration mode to alert about Silva's wife message). When Silva is at home or somewhere else — e.g. at an Airport —, the Configuration Management Service will be immersed in an different overall context. In such cases, formalization may help to describe and understand how different contexts form a specific combined view.

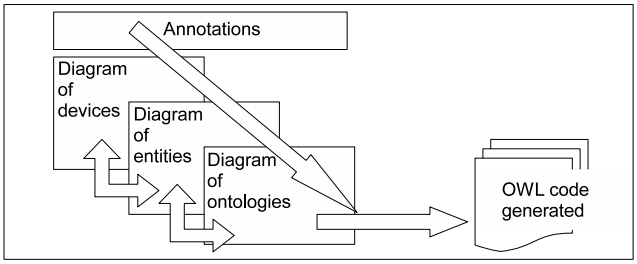

Figure 4: Overview of the environment for design and implementation of ubiquitous systems.

Let's assume that at a certain time Silva is visiting UFF with several other researchers and, as usual, 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 enables SMP_{Silva} to connect to the network, and using the current GPS data, queries a location service to find out that its owner (Silva) is at UFF. 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. 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. 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.

3.2 The Modules

The environment of ubiquitous systems design and implementation is composed by the following modules as shown in figure 4:

Diagram of Devices and Applications. The diagram of devices and applications specify all computational devices (pieces of software/hardware) that compose the ubiquitous system and the kind of information that is exchanged among them. Each device of an ubiquitous systems is represented by a node of the diagram, which is linked to other nodes by an arrow labelled with the information that is to be transmitted from a device or application to another. Examples of devices and applications are a mobile phone and a personal agenda, which can exchange information as, for example, the date of a meeting.

EXAMPLE 2. *We show in figure 5 a simple diagram of devices that considers several devices: the Ambient Manage-*

ment Service (AMS) that detects the presence of a person carriyng a smart phone and registers its code; the Ubiform Project Agenda (UPA), that informs the members of UbiForm Project the arrival of a member; the Configuration Management Service (CMS), that, when receiving a signal from AMS, configures the phone mode to silence or vibration if the user is in an activity room; a Personal Agenda (PA), that contacts a project schedule and signals the user the beginning of an activity, and, finally, smart phones and workspaces.

Diagram of Entities. The entities of an ubiquitous system are components that produce a reaction. For example, a Professor, that must be identified when arriving at work is an entity. In this situation, the device that captures the presence of the professor is the context wherein the professor is emmersed at that time. But a context is also an entity, as it certainly produces a reaction in another device. The algebra of contextualized entities gives strong support on this flexibility of changing role: adopting a uniform representation for entity and context, and also because links compose associatively, one context can act as entity of a different context. This latter acts as (meta) context of the entity. It is also possible for an entity to have several contexts (several links with the same domain) or for a context to contextualize several entities (several links with the same codomain). These situations can be extended indefinitely forming a net of entities and contexts, that together provide an understanding of an object or situation. The algebra of contextualized entities formalizes the manipulation of entities providing operations to generate new entities wherein the devices can extract the necessary information to work on.

EXAMPLE 3. *Figure 6 is part of a diagram of entities that relates personal information of Professor Silva (in the ontology Silva) with his physical position at PUC-Rio, as a member of UbiForm Project (in the ontology UbiForm Project).*

Annotation pane. The annotations can be defined, deleted or associated to different components of diagrams or code during the process of specification.

EXAMPLE 4. *According to the scenario, the diagram of entity of figure 6 is related to the fact that if Silva's wife sends him a message during the meeting, the phone should silently vibrate. This condition, that will appear in the whole description, as an axiom of the diagram of ontologies can be annotated while spcifying the diagram of entities to not be forgotten later. The annotation could be:*

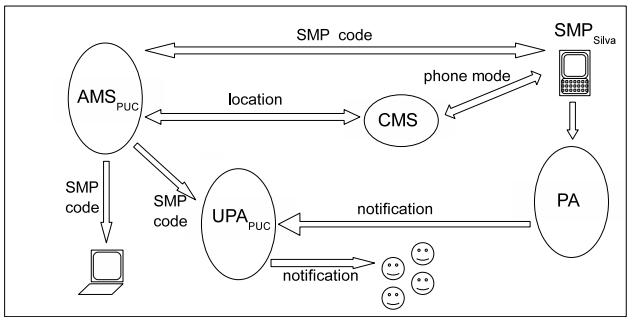

Figure 5: A diagram of devices.

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

Diagram of Ontologies. The diagram of ontologies provides a detailed view of the diagram of entities, where each entity is described by an ontology and the links are *consistently defined*. In the algebra, a link is consistently defined if it maps components of the entity (domain ontology) into the context (codomain ontology) in a way that the hierarchy of concepts and relations of the entity are reflected in the codomain, and the axioms of the entity, when properly translated to the vocabulary of the context, hold for the context.

EXAMPLE 5. *We consider a situation in which information coming from one context enables decisions about an entity in a different context. For instance, 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 an annotation that would set an access permission in a ubiquitous regulation service, such as in [9]:*

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

Considering the base square of the diagram of entities in Fig. 12, the mediator $Prof.\ Silva$ of the context integration $UFF \overset{AMS}{\longleftarrow} Prof.\ Silva \overset{AMS}{\longrightarrow} PUC$ 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 available to be used by AMS.

3.3 Relating the Modules

The modules described above are related in the following way: each device of the diagram of devices correspond to a diagram of entities, for example, figures 11 and 12 show steps of the construction of the diagram of entities for AMS, a device of figure 5. But, in the diagram of entities, details of entities and their connection are hidden. These details appear in the diagram of ontologies, where entities specifications and their connections are sufficiently detailed in order to make possible consistency tests. Respecting this correspondence the annotation pane is able to relate an annotation in a diagram to the corresponding component in

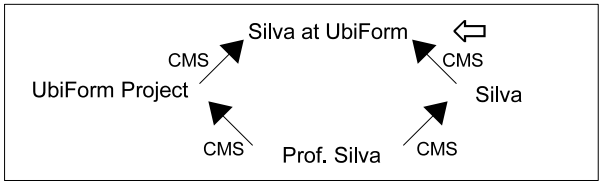

Figure 6: Diagram of entities: CMS considers personal information about Silva and his physical position at the UbiForm.

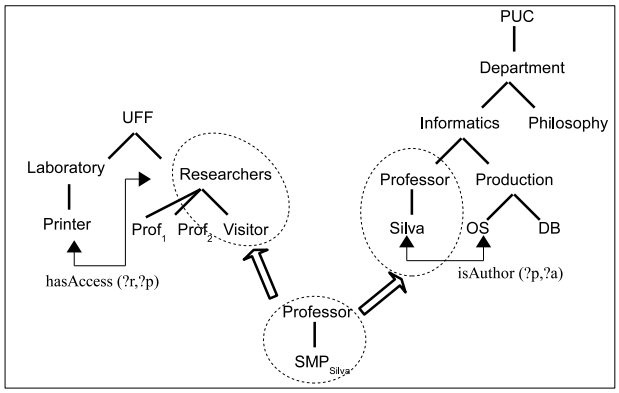

Figure 7: Diagram of ontologies: Alignment of UFF and PUC under the mediation of $Prof.\ Silva$. The mediator captures the fact that Silva is a professor and properly map this information in the ontology of UFF.

each of the other diagrams. Moreover, the process of specification can be performed on the three windows of diagrams concurrently such that a change in a window may produce effects on the other two. At the same time, annotations associated to components of the diagrams can be made at any time. These annotations can be accessed from the related components of the other diagrams or from the related portion of the generated code.

For the sake of simplicity, we describe the process of specification in linear way.

Starting by the definition of the diagram of devices, which presents the higher level of abstraction, the user sketches the flow of information among the several devices that compose the system. When inserting a device in the transition diagram, a double click in the mouse transfers the user to the diagram of entities of that device. In this diagram, the

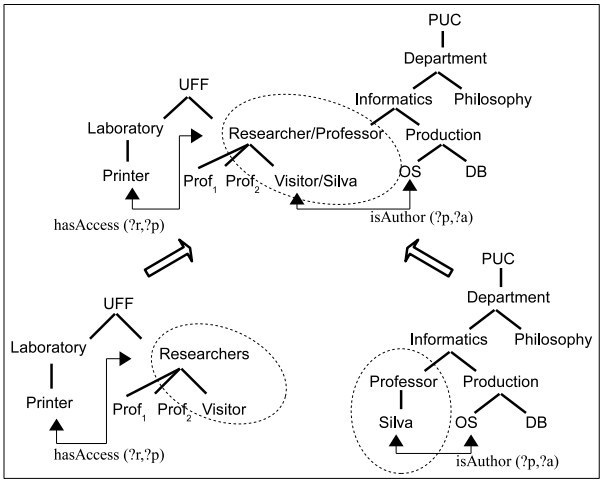

Figure 8: Diagram of ontologies: The context integration of the alignment of figure 7. The relation *hasAcces(Researcher, Printer)* holds for Professor Silva and Printer and information about Professor Silva's production is avaiable.

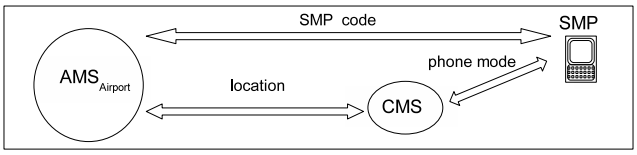

Figure 9: Diagram of devices: Professor Silva at the airport.

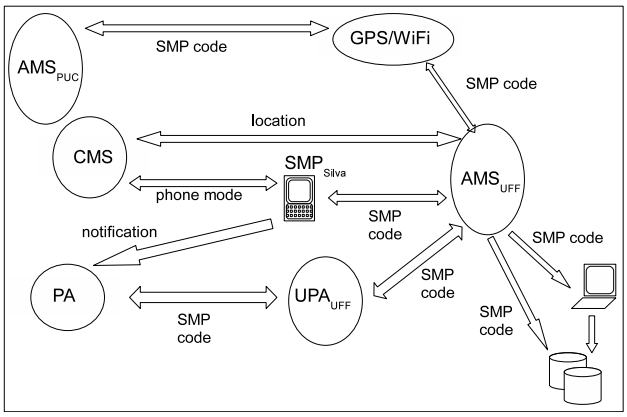

Figure 10: Diagram of devices: Professor Silva at UFF.

user constructs the body of information over which the device must work. A diagram of ontologies for each entity of the diagram of entities can be generated as long as information is being added to the diagram of entities, guided by the algebra of contextualized entities. Additional information to complete ontologies and ontology links may be inserted in a latter moment, and validation of links can be semi-automatically done. From the diagram of ontologies it is possible to derive a partial code in OWL or a similar language.

4. A CASE STUDY ON UBIQUITOUS COMPUTING

We present an overview of the formalization of the scenario presented in 3.1. Since the whole formalization would exceed the space limit, the objective of this section is just to give an idea of the integration between the diagrams that are constructed in each module.

4.1 Diagrams of Devices

Figures 5, 9 and 10 compose the diagram of devices of the scenario described in subsection 3.1. As commented in example 3, figure 5 shows AMS, CMS, UPA and PA that interchange information as a result of the perception of the presence Professor Silva. Two of these devices are applications running on Silva's smart phone: the CMS, that configures the phone mode to silent, alarm or vibration according to the room or activity where Professor Silva is engaged, and PA, a personal agenda that must be synchronized with the project agenda. The other two devices (AMS and UPA) are applications running at the environment and play the role of monitoring information about the environment and notifying the SMP applications. Figure 5 pictures the ex-

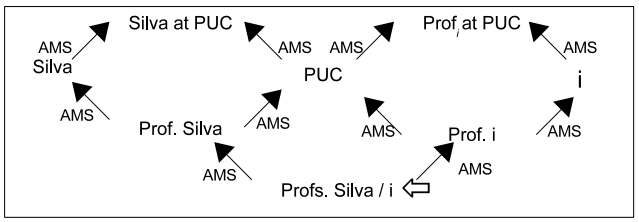

Figure 11: Diagram of entities: Integration of several professors.

changing of information concerning the second paragraph of subsection 3.1, when Professor Silva arrives at PUC.

In figure 9 we consider just AMS (running at the airport) and CMS (running on Professor Silva' smart phone) and formalize the situation where there is no restriction to the phone alarm, thus the phone configuration must be restored.

Finally, as in the fourth paragraph of subsection 3.1, we picture, in figure 10, the diagram of devices that formalize the arrival of Professor Silva at UFF. In this case, it is performed a communication between AMS, at PUC, and AMS at UFF, via GPS/WiFi. Through this communications, Professor Silva is recognized at UFF and resources can be allocated according to his preferences and permissions.

4.2 Diagrams of Entities

This diagram will compose a coherent and minimal body of information over which the device must work. For example, for AMS it is necessary to combine personal and professional information of Professor Silva in order to use personal preferences to configure professional environment.

The Ambient Management Service (AMS). Figure 11 concern the situation where AMS informs other members of Silva's team about his arrival. For any member $Prof_i$, a context integration $i \overset{AMS}{\longleftarrow} PUC \overset{AMS}{\longrightarrow} Prof_i$ ($Silva \overset{AMS}{\longleftarrow} PUC \overset{AMS}{\longrightarrow} i$) is generated resulting the ontology $Prof_i$ *at* PUC ($Silva$ *at* PUC), that combines personal and professional information for each i. The entity integration of each $Prof_i$ and $Prof\ Silva$ under the context of PUC (lower square of figure 11) will make the connection among the i professors of PUC and Professor Silva. Over this ontology, AMS has access to the integrated information about all professors.

Later, Professor Silva is visiting UFF, where he is registered as a visitor researcher. Within the context $SilvaAtUFF$ that results from integration $Silva \overset{AMS}{\longleftarrow} Prof.Silva \overset{AMS}{\longrightarrow} UFF$, 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. Thus, information about Silva's status at PUC must also be taken into account. The context integration $UFF \overset{AMS}{\longleftarrow} Prof.Silva \overset{AMS}{\longrightarrow} PUC$ 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 Figure 12). 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 Figure 12). Note that Figure 12 also pictures a combined integration: the collapsed union of the contextualized entities $UFF \rightarrow SilvaAtUFF$, $PUC \rightarrow SilvaAtPUC$ mediated by $Prof.Silva \rightarrow Silva$.

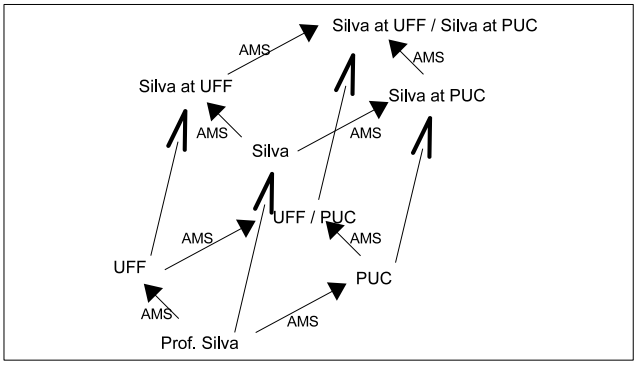

Figure 12: Diagram of entities: Each face of the cube shows a context integration. The complete cube is the collapsed union of the contextualized entities $UFF \rightarrow SilvaAtUFF$, $PUC \rightarrow SilvaAtPUC$ mediated by $Prof.Silva \rightarrow Silva$.

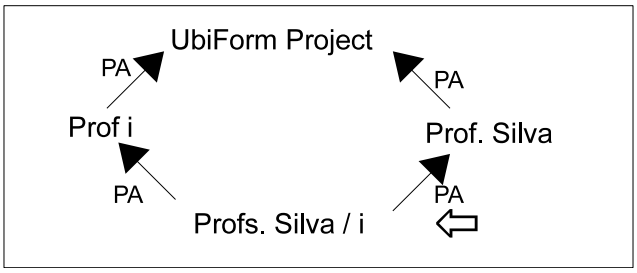

Figure 13: Diagram of entities: Entity Integration that results in the synchronization of several smart phones Personal Agendas with respect to the UbiForm Project Agenda.

The Personal Agenda (PA). The personal agenda of Silva's smart phone contacts the UbiForm Project Agenda to be notified about scheduled activities. The entity integration $Prof\ Silva \overset{PA}{\longrightarrow} UbiFormProject \overset{PA}{\longleftarrow} Prof_i$ 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 (figure 13).

The Configuration Management Service (CMS). The configuration management service 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 Silva's personal information in order to properly configure his phone alarm.

A context integration $UbiFormProject \overset{CMS}{\longleftarrow} Prof.\ Silva \overset{CMS}{\longrightarrow} Silva$ results in a context $SilvaAtUbiForm$ which combines personal information about Silva and the present UbiForm activity in which he is involved (figure 6). Similar situation occurs when Silva is somewhere else, e.g. as at the airport. The context integration $Airport \overset{CMS}{\longleftarrow} Prof.Silva \overset{CMS}{\longrightarrow}$

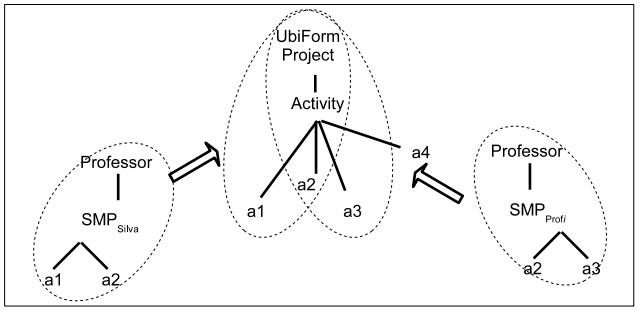

Figure 14: Diagram of ontologies: The coalignment of SMP of Professor i and SMP of Professor Silva with respect to the agenda of the UbiForm Project.

Silva results in the context *SilvaAtAirport* wherein the CMS can configure his phone alarm according to his contextual preferences.

4.3 Diagrams of Ontologies

The diagrams of ontologies are a refinement of the diagrams of entities, were each entity appears described as an ontology and the connections are links between ontologies (that is, they preserve ontology properties).

Detailed description of the whole scenario would exceed the space limitation and is not the objective of this section. In example 5 we showed how diagram of ontologies preserve structure and relations. In this section we selected a diagrams of entities to illustrate how the integration can filter information in order to affect just a selected set of entities. We consider the situation, where the Personal Agenda of Silva's smart phone contacts the UbiForm Project Agenda to be notified about events.

Diagram of Fig. 13 pictures this situation, showing the integration of SMP of Professor i and SMP of Professor Silva under the context of the UbiForm Project. Figure 14 shows the coalignment of SMP of Professor i and SMP of Professor Silva with respect to the context of the UbiForm Project. Figure 15 shows the resulting entity, in which appears only the events that both take part.

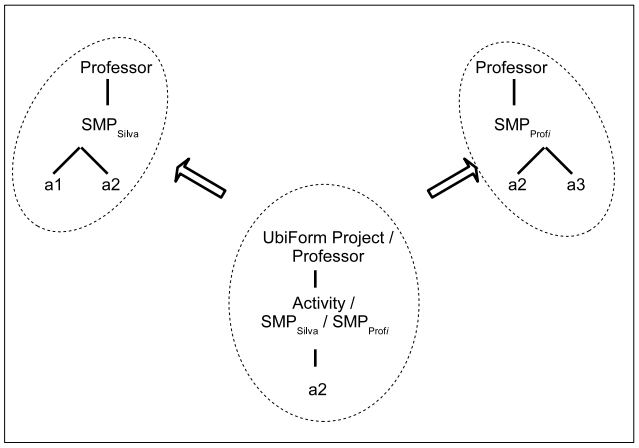

Figure 15: Diagram of ontologies: Integration of agendas: Silva and Professor i will be present at Event 2.

5. CONCLUSION

This paper proposes an environment to support the use of the algebra of contextualized ontologies for the specification context modelling in ubiquitous systems. The algebra is designed to reduce the gap between a system's implementation and formalization, so as to minimize some difficulties faced by programmers when writing formal specifications. The environment is designed to highlight and make available to the programmer all the facilities of the algebra, which is based on the principle of an homogeneous and independent description of entities and contexts and a representation of their semantics as a network of relationship. The algebra adopts ontologies to represent entities and contexts and the links between an entity and a context are the ones that inform, in each particular situation, which ontology plays the role of an entity (i.e. the domain of the link) and which ontology plays the role of context (i.e. the codomain of the link). Since the links of the network compose associatively, a context can act as an entity of a wider context, which thus plays the role of a meta-context of the former context. It is also possible for an entity to have several contexts (several links with the same domain) or for a context to contextualize several entities (several links with the same codomain). Such relationships can be extended arbitarily, forming a network of entities and contexts, which, as a whole, provide an understanding of an object or situation.

The environment emphasises these features by splitting the algebra in several levels of abstractions, each one represented by specific diagrams. These multiple levels of abstractions are integrated and can be modified at any time. Annotations can be associated to elements of one diagrams, and then, through transformations, can be propagated and refined according to the level of abstraction of each diagram. These features were shown through an example of a ubiquitous computing application.

The operations of the algebra of contextualized entities serve the purpose of co-relating and integrating information among several levels of a system's representation. The ultimate goal is to be able to construct a complete and minimal description of a system upon which a component of an ubiquitous system can reason.

6. REFERENCES

[1] I. Cafezeiro and E. H. Haeusler. Semantic interoperability via category theory. *Conferences in Research and Practice in Information Technology*, 83:519–533, 2006.

[2] I. Cafezeiro and E. H. Haeusler. Ontology and Context. In *Proceedings of the Sixth Annual IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 417–422, 2008.

[3] I. Cafezeiro, J. Viterbo, A. Rademaker, E. H. Haeusler, and M. Endler. A formal framework for modeling context-aware behavior in ubiquitous computing. *3rd International Symposium on Leveraging Applications of Formal Methods, Verification and Validation. CCIS: Leveraging Applications of Formal Methods, Verification and Validation*, 17, 2008.

[4] H. Ehrig and B. Mahr. *Fundamentals of Algebraic Specification 1: Equations and Initial Semantics.* Springer-Verlag, 1985.

[5] E. H. Haeusler and J. Meseguer. MEFIA Protem/NSF : Mathematical and Engeneering Foundations for Interoperability via Architecture. Technical report, PUCRio, S.R.I International, 2000.

[6] C. Julien and G.-C. Roman. Egocentric context-aware programming in ad hoc mobile environments. In *SIGSOFT '02/FSE-10: Proceedings of the 10th ACM SIGSOFT symposium on Foundations of software engineering*, pages 21–30, New York, NY, USA, 2002. ACM.

[7] Y. Kalfoglou and M. Schorlemmer. Ontology mapping: The state of the art. In Y. Kalfoglou, M. Schorlemmer, A. Sheth, S. Staab, and M. Uschold, editors, *Semantic Interoperability and Integration*, number 04391 in Dagstuhl Seminar Proceedings. Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany, 2005. <http://drops.dagstuhl.de/opus/volltexte/2005/40> [date of citation: 2005-01-01].

[8] A. Lopes and L. Fiadeiro. Context-awareness in software architectures. In *EWSA 2005: Proocedings of the 2nd European Workshop in Software Architecture*. Springer-Verlag, 2005.

[9] J. Viterbo, M. Endler, and J.-P. Briot. Ubiquitous service regulation based on dynamic rules. In *Proceedings of the 13th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS 2008), Belfast*, pages 175–182. IEEE Computer Society Press, 2008.