A Semantic Database Framework for Context Management in Heterogeneous Wireless Networks

Mehdi LOUKIL*, Takoua GHARIANI*,
Badii JOUABER*† and Djamal ZEGHLACHE*†

(*) Institut TELECOM / TELECOM SudParis
(†) CNRS, UMR SAMOVAR
{Mehdi.Loukil, Takoua.Ghariani, Badii.Jouaber, Djamal.Zeghlache}@it-sudparis.eu

Abstract. Recent developments in computer science and networking raise the need to develop new families of applications that are sensitive to user and ambient contexts. The objective is to offer personalized services that can be adapted to users' needs within heterogeneous, complex and dynamic environments. This requires enhanced and generic solutions for context representation and retrieval as well as for the interactions between heterogeneous system components. In this paper, we propose a conceptual model and a software framework using semantic ontologies to build semantic context databases. This approach is then applied to the heterogeneous wireless networks' use case. Contextual information is implemented using first order logic and OWL to promote expressiveness and interoperability. The different steps to build the framework are detailed. Measurements and performance results are then provided and enhancements with concept/instance separation approach are discussed.

Keywords: semantic database, ontology, OWL, context management, heterogeneous wireless networks.

1 Introduction

With the continuous evolutions of networking and information systems and technologies, services and network providers as well as end-users are facing increased complexities to deliver/access personalized and adapted services. Indeed, a multitude of heterogeneous and competing solutions are available nowadays for service delivery and execution environments. However, it is still very hard to adapt existing services, initially developed for homogeneous environments (mono operator, mono technology, mono architecture, etc), to make them available for different settings and technologies. In addition, with mobility and nomadism, users expect to get seamless access to their personal and professional services (including multimedia for business and infotainment). Automated service (networking and applicative) discovery, composition and adaptation to users and systems' contexts are then needed.

To achieve these goals, new functionalities are required in future information and networking systems such as context awareness and automated reasoning. Contextual information should be available and accessible in comprehensible and unified formats to allow systems reacting without (or with minimum) human interventions (i.e. self-adapt). Context-aware computing is widely accepted as a suitable paradigm for the development of applications for mobile and pervasive environments. Its main purpose is to allow dynamic adaptation of applications and services in order to guarantee an adequate usage (comfortable and optimized) of devices and network resources, and to properly handle runtime applications' requirements.

These reasons motivate the design of a generic and open context management system that supports environments heterogeneity.

The objective of this paper is to propose and evaluate the design of an ontology based context management framework to support pervasive computing over heterogeneous networking environments. We consider solutions for loosely-coupled architectures (mostly used to interconnect networks belonging to different operators). A semantic data base is then proposed for context management over such environments to provide extensibility, expressiveness and flexibility for the development of context-aware services.

The remainder of this paper is organized as follows. After the problem statement and the review of the state of the art, the proposed context management framework and the context modeling approach are presented. The implemented testbed and evaluation results are then provided. Finally, proposals for performance enhancements are presented and discussed before conclusions.

1.1 Problem Statement

Within smart systems, users are expecting to come across automated and personalized services. This requires advanced mechanisms for service adaptation and for the exchange and the analysis of users' contexts, profiles and preferences (i.e. reasoning). In current systems, *Request/Answer* mechanisms represent basic interactions between users' equipments and the system environment to exchange contextual data. Hence, there is an essential need of designing a global context manager that can be integrated, for instance in an overlay virtual system, and making available the required knowledge to ease service personalization and adaptation. On the other hand, ontologies can provide rich and unified semantic descriptions of complex systems such as nowadays heterogeneous networking environments,

The issue in the paper is to build a context manager using ontologies. Individuals (instances of the ontologies) are added to the semantic description to setup a knowledge database with relations, rules, context and profile information. The proposed system should be able to detect major changes in users' profiles or needs, or more generally within the entire system context (ambient conditions) to ease the

adaptation and the selection of the most suitable service mode and to deliver it to the user at the session initiation or at runtime.

We aim in this paper to analyze the feasibility and the performances of such advanced systems within rich and complex heterogeneous networking environments. We present and discuss two conceptual approaches that offer two different levels of performances.

1.2 Related Work

In the literature, different studies and projects tackled some of context, context-awareness, semantic resources description and ontology querying aspects.
From the various refinements in the definition of the term context awareness, Dey's definition have become widely accepted and adopted to provide a consistent understanding of the subject studied by many researchers. Dey [1] defined context as '*Any information that can be used to characterize the situation of an entity. An entity is a person, place or object that is considered relevant to the interaction between a user and an application, including the user and application themselves*'. In [2], authors considered the use of Semantic Descriptions to view services and devices as functional components. Such services may be introduced within an open service community, and consequently located, possibly added to a workflow models, and subsequently invoked. A framework to describe devices on the web through ontologies is proposed in [3]. The advantages of such frameworks include the ability to extend and adapt the vocabulary used to describe services, to utilize existing concepts defined in alternate ontologies, and to harness the inferential benefits of logical reasoning over such descriptions. Authors also compared their proposal to the semantic web service abstract models introduced in [4]. Context awareness systems are introduced in [5] where authors discussed and proposed possible architectures to integrate context management on existing systems.

The Aura project [6] proposed a context aware system to provide proximity services to mobile devices. Localization mechanisms are based on WLAN power measurements. Authors propose to boost the "limited" capacity of mobile devices through the use of servers, on the infrastructure, on which contextual information are stored and analyzed. Although the basic idea of this project is to propose localization based services, it does not solve heterogeneity and autonomy aspects.

The DoCoMo laboratory [7] proposed a context management Framework to optimize vertical HO between different network technologies of a given operator. The HO decision (taken on the mobile side) is based on collected contextual information on a centralized server side: "the context collection point". The main advantage of this proposal resides in the introduction of autonomy on mobile devices. However, it does not consider heterogeneous environments since it is only intended to mono-operator managed networks. In addition, it does not alleviate mobile devices from hard processing.

Many approaches have been proposed for querying ontologies. In the context of OWL reasoning, Pellet [8] is a complete OWL-DL reasoner with extensive support for reasoning with individuals and user-defined datatypes. Queries are specified in the RDQL [9] or the SPARQL [10] query languages, which have a syntax similar to SQL, but built around a set of conjuncted triple patterns. RacerPro [11] is another OWL-DL reasoner.

2 Proposed Framework

The purpose of this solution is to generalize the use of context management in heterogonous networking environments in order to support high flexible service provisioning. In addition, an ontology is proposed to describe and store environment and context information.

Although, there is some related work concerning the description of devices, so far there has been no collective effort to come out with a formal framework to describe entities and devices to facilitate semantic service discovery, adaptation and composition. In this paper, an OWL-based ontology is proposed. Its aim is to provide a formal framework to describe system entities and devices to support effective and personalized service discovery and adaptation.

To automate context analysis and reasoning, the virtualization framework proposed in [12] is used. It is a conceptual control level (see Figure 1), composed of two virtualization layers. It uses software delegates, denoted *Software Avatars* to enables reasoning on behalf of physical and logical entities within the system. It aims to ease the management of heterogeneous entities including users, terminals, services, networks, service providers, etc.

2.1 High level description

Virtualization is achieved through the use of *Software Avatars*. These are software entities, with communication and reasoning capabilities, able to act on behalf of the entities they represent. The two virtualization layers are defined according to the dynamicity of contextual information they manage. The first virtualization level is defined to make reasoning on frequently changing contextual information. The second level is defined to make reasoning on static contextual information. It has a global view on the entire system and is responsible of orchestrating the first level.

In this paper, we only focus on context management. Within the second abstract level, it consists of a *semantic data base* responsible of representing, storing, gathering and retrieving contextual information.

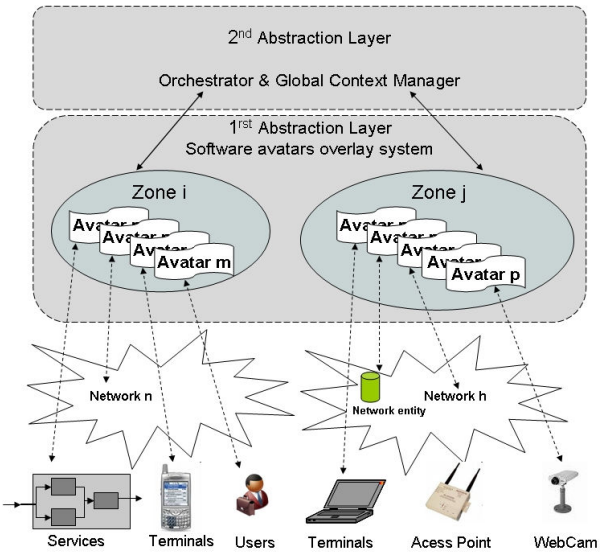

Figure 1: A Two-layers Virtualization Overlay Architecture [12]

2.2 The context manager framework conception

The second abstraction level contains a global Context Manager and an Orchestrator. These two components have a global view of the system. They offer a unified representation through ontologies allowing reasoning and inferring.

In our opinion, an ontology-based data representation can bring efficiency in information exchanges between running applications and system entities. Adding semantic information to real data is very useful to enable and ease automated processing. On the one hand, the semantic aspect, introduced by ontologies, can enhance the retrieval and access to the required information when needed. On the other hand, the distributed software avatar-based architecture enables the interworking and exchanges within open and heterogeneous environments. At this level, the context manager is composed of the ontology for describing physical and logical system entities as well as reasoning and inferring modules. These use the ontology to extract contextual information (instances of the ontology concepts, which can be separated from the concept description) as response to context requests.

A well conceived ontology makes possible to have a global view on the environment it describes. It may be considered similar to a well-defined database schema [13].[13] It enables automated reasoning and inferring modules. Another advantage is to automate communications between system components including the Avatars, the Orchestrator and the correspondent physical entities (see Figure 2).

Among the advantages of such a framework are the abilities (i) to extend and to adapt the vocabulary used to describe services, (ii) to utilize existing concepts defined in alternate ontologies, and (iii) to harness the inferential benefits of logical reasoning over such descriptions. Such benefits are necessary within dynamic and evolving environments (particularly with respect to mobile computing), where no assumptions can be made about the availability of any given service, device or network.

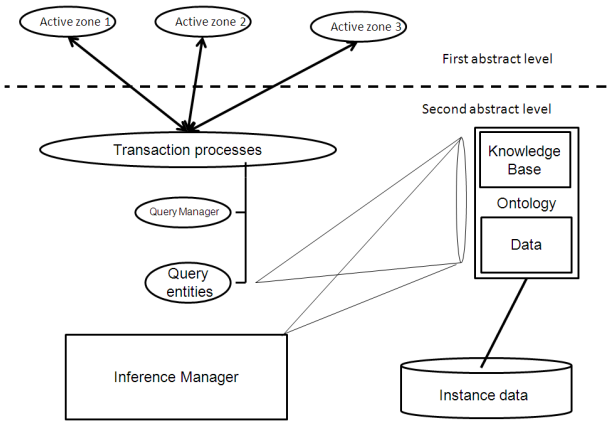

Figure 2: Second abstract level overview [12]

2.3 Ontology conception

Describing services using ontologies is superior to using other forms of data structures, because the former method provides a structure that makes it possible to reason about and derive knowledge from the given descriptions. By using ontologies, the relationships between entities can be more clearly expressed and they allow for better reasoning.

When a user invokes a service (e.g. a video streaming/conference service, printing service, scanning service, etc) some level of details about it will be required for service mode selection purposes (adaptation level). For example, in the case of a video streaming service, the different proposed quality levels of this service might be useful to select the best service mode according to the user' device capabilities and to the available network bandwidth. Such information, related to the service appropriateness could be included along with the service/device/network description itself (one ontology). However, separated descriptions of devices and services promote easy to use, readability and reusability.

In some cases of service composition where adaptation is involved, it will be necessary to reason about the capabilities of available services in order to determine a negotiator/broker platform, where the execution and coordination of the elementary

services take place. The broker platform selects elementary services based for instance on their proximity to the user device and service capability. In such cases the ontology becomes useful in describing the capabilities of the devices/services available within the network/environment.

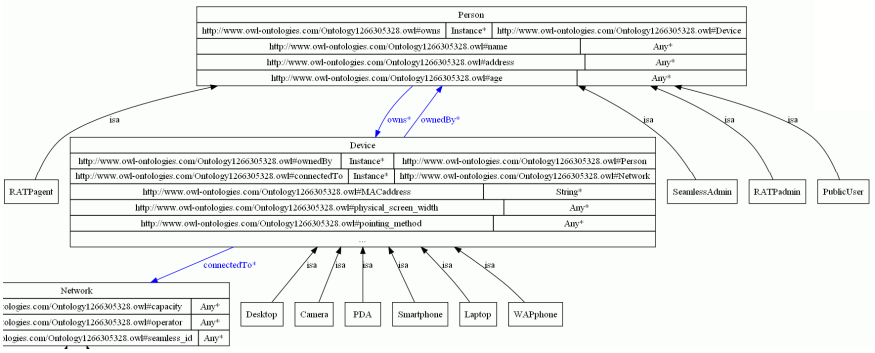

Figure 3.a. **Illustration of some concepts from the proposed ontology, which is mostly self-explanatory.**

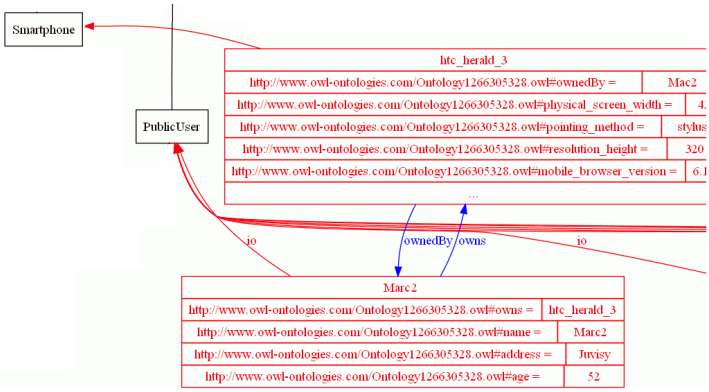

Figure 3.b: **Illustration of some concept instances from the proposed ontology, which is mostly self-explanatory.**

Device Description

The information related to a device is logically divided into some classes depending on the type of awareness they provide: namely Device Description, Hardware Description, Software Description, Device Status and Services. Device Description contains basic information related to a device such as the device identifier, Vendor details and the Model of the device. Hardware Description contains the details about the hardware resources of the device, the details of its CPU,

connection types to the networks, screen characteristics, and memory. Software Description contains the details of the Operating System of the device where relevant. The Device Status contains the details of its location, CPU usage and the power (method of power supply, whether its battery or mains and the remaining power level). The details of power supply and power level becomes important when it is necessary to determine the resource capability of a device. Location is required when service (application or network service) selection needs to consider the location of the device in choosing the right service.

Service description

The Service class provides the information about the service(s) hosted on the device concerned. OWL-S could be potentially used to describe these individual services. There is a *1:n* relationship between the Device class and the Service class. For example if a particular device has a printing service, video conference service and a camera accessible service, there will be three Service classes in the device ontology for that particular device. The Device Ontology is intended to provide a general framework to describe any type of device (e.g. a software avatar within the system). But to describe specific types of devices more precisely, the concept of class hierarchies can be used. A hierarchy of sub-classes can be constructed, that inherits from the Device class to provide an effective device categorization. For example there can be a Camera sub-class that inherits from the device class and builds on additional properties (such as Camera resolution etc.) as necessary to effectively describe Cameras. In the case where a device does not fall into any of the available categories, or when it is not clear to which category a device belongs to, it could be specified as an instance of the Device class itself and thereby avoiding the use of the hierarchical classification.

Network description

The Network Description class contains basic information related to a Network such as network bearers, capabilities (bandwidth etc.), distribution and availability (location etc.).

The Network Status contains the details of its load, interference with other network signals, security levels and methos, etc. There is some relationship between the Device class and the Network class. For example when Device has the capabilities to connect to Network or to save past connections/sessions, relationship between these classes can be added.

User profile description

The User profile class provides the information about the physical person (system' user). The information related to a user is logically divided into some classes depending on the type of information they provide: namely User General Information, User preferences and Requirement, particularly in terms of network access technology, network availabilities, services features such as video quality in case of a video streaming session, etc. This is enabled by a set of predefined relationships and restrictions between users' profiles and network, devices and services features.

2.4 The context transaction process

The context transaction process consists in a set of functional blocks that translate both messages received from context manager users, as requests, and their respective responses from the data bases. This transaction process is a reasoned (PELLET) enabled component that is able to access the ontology data.

2.5 Work profile

Work profile consists in the combination of the static context data provided by the semantic database (second abstract level) and the dynamic one gathered from physical entities (devices, networks, and services).

3 Testbed and evaluation works

In order to experiment and evaluate the proposed concepts and designs, we extended the real testbed proposed in [12] using a JADE MAS (Multi-Agent System) [15]. Extensions were necessary to support an ontological description of the environment, namely networks' features, equipments' constraints and services' requirements. We used Protégé [14] to design the ontology. Figure 4 presents the context manager implementation design.

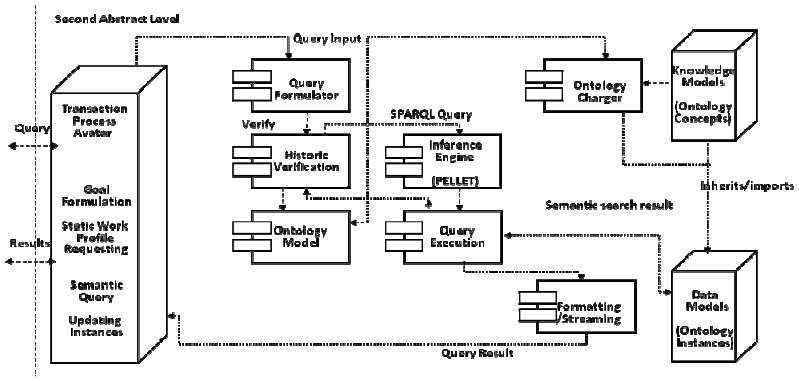

Figure 4: **Context Manager Design.**

In this first design, Context Data are stored on the ontology (descriptions and Instances). We used The Pellet Reasoner [8] to fill the ontology model and execute SPARQL (a recursive acronym standing for SPARQL Protocol and RDF Query Language) query entities. To format query entities, we developed a specific java tools (using Jena and OWL-API). Results of the queries are delivered to the corresponding *Avatar* (that formulated the query).

This first ontology design consists of a one OWL file containing concepts, relations, proprieties and datatypes semantic description and individuals. Performance measurements (response times) are provided in Table 1. Results are obtained for complex queries: extracting user profiles and their device characteristics (around 15 context data elements). Presented response time values are obtained by averaging the results of 500 different queries.

Table 1. Time response Evaluation

Ontology size (instances)	20	100	200	400	1000	1600	2000	4000	6000	8000
Average Response Time (in ms)	65,76	109	127	191,5	395,7	568,9	717	1330	1965,6	2582

As expected, this first implementation provided the projected functionalities. However, the measured response times grow exponentially with the ontology size and are not adapted to the targeted use-case. To enhance these performances, we decided to go for separating the context data from the semantic description. This new solution consists in two OWL files. The first one contains data (individuals), the second contains the semantic descriptions. This separation should reduce the file sizes and speed up the access time.

Figure xxx depicts the obtained results and provides comparisons between the two conceptual approaches. We observe enhancements between the one-file and the split-files solutions.

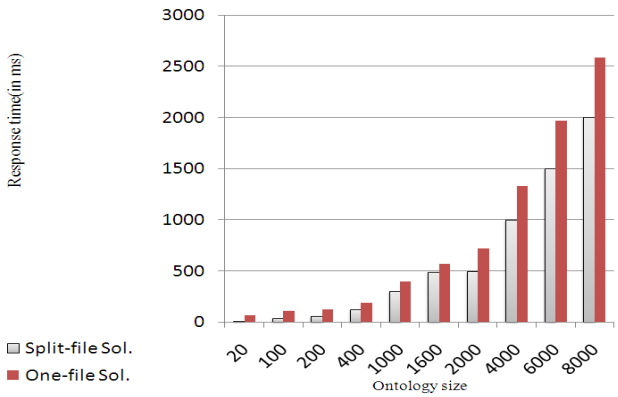

Figure 5: Response Time Optimizations after Splitting.

5 Conclusions and future works

In this paper, we proposed a semantic context management framework for heterogeneous wireless networks based on ontologies. Actually, this work consists in the conception of an ontology and the definition of a context framework and its components (i.e. a transaction process and a query manager). These components enable the extraction and the update of the context data within the ontology repository. We then proposed two designs and implementations (one-file and split-files ontology designs) for the context semantic database. Early performance evaluations show significant differences in terms of response time between these two solutions.

New efforts are ongoing to investigate new approaches and enhancements to optimize availability and response times especially for real time tasks like vertical handover mechanisms and service adaptation.

References

[1] Dey, A. K., Providing Architectural Support for Building Context-Aware Applications, PhD. Thesis, Georgia Institute of Technology, 2000.

[2] Ayomi B., Terry P., David de R., Gary C., "An Ontological Framework for Semantic Description of Devices", 09/2004

[3] McGuinness, Natalya F. Noy and Deborah L. Ontology Development 101: A Guide to Creating Your. Stanford, CA: Stanford University, 2002. 94305

[4] Berners-Lee, T., Hendler, J. and Lassila, O. The Semantic Web, Scientific American, May 2001. Also http://www.w3.org/2001/sw/

[5] Dey, A. K., Providing Architectural Support for Building Context-Aware Applications, PhD. Thesis, Georgia Institute of Technology, 2000.

[6] Project Aura: toward distraction-free pervasive computing (Garlan, D. Siewiorek, D.P. Smailagic, A. Steenkiste, P. Carnegie Mellon Univ. Pittsburgh, PA).

[7] A Framework for Context-aware Handover Decisions (Prehofer, C. Nafisi, N. Wei, Q. DoCoMo Commun. Lab. Europe, Munich, Germany)

[8] Pellet, http://www.mindswap.org/2003/pellet/

[9] Andy Seaborne, “RDQL A query language for RDF”, W3C Member Submission 9 January 2004, http://www.w3.org/Submission/RDQL/

[10] Eric Prud'hommeaux, Andy Seaborne (eds), “SPARQL Query Language for RDF”, W3C Working Draft 4 October 2006, http://www.w3.org/TR/rdf-sparql-query/

[11] RacerPro, http://www.racer-systems.com/

[12] M. LOUKIL, B. JOUABER, A two-Layered Virtualization Overlay System using Software Avatars, ISCC 2010 Italy.

[13] Salt Lake City, Utah, A Coevolution Approach for Database Schemas and Related Ontologies, June 22 2006, ISBN: 0-7695-2517-1.

[14] Protégé, a suite of tools to construct domain models and knowledge-based applications with ontologies.
http://protege.stanford.edu/overview/index.html

[15] Rimassa G., Bellifemine F., Poggi A., "JADE - A FIPA Compliant Agent Framework", PMAA`99, p. 97-108, London, April 1999