Hybrid Approach to Collaborative Context-Aware Service Platform for Pervasive Computing

Dejene Ejigu
Laboratoire LIRIS-UMR-CNRS 5205, INSA de Lyon,
7 avenue Jean Capelle, 69621 Villeurbanne cedex, France
Email : dejene.ejigu@insa-lyon.fr

Marian Scuturici
Laboratoire LIRIS-UMR-CNRS 5205, INSA de Lyon,
7 avenue Jean Capelle, 69621 Villeurbanne cedex, France
Email : marian.scuturici@insa-lyon.fr

Lionel Brunie
Laboratoire LIRIS-UMR-CNRS 5205, INSA de Lyon,
7 avenue Jean Capelle, 69621 Villeurbanne cedex, France
Email : lionel.brunie@insa-lyon.fr

***Abstract* – The behavior of pervasive applications depend not only on their internal state and user interactions but also on contexts sensed during their execution. In this work, we present a collaborative context-aware service platform based on hybrid context management model (enhanced CoCA). It performs reasoning and decisions based on context data, context semantics, and related rules and policies. Such data is organized into a hybrid context management model (HCoM). The platform also introduces a neighborhood collaboration mechanism to facilitate peer collaboration in order to share their resources. We have developed an initial prototype of the platform. Our preliminary test shows that the platform is a promising data independent development environment for pervasive context-aware applications and it possesses good standard of scalability.**

***Index Terms*: CoCA, context management, reasoning, context awareness, pervasive computing**

I. INTRODUCTION

The emergence of a computing model for mobile ad-hoc networks in pervasive environments, the wide spread of pervasive enabling technologies and the availability of computing enabled handheld appliances like smart phones and personal data assistances make computing more distributed in such a way that computing could be with the user every where and every time. The growing interest in the area and the concern that users should not be disturbed by such devices while they are on their regular duty has introduced several research challenges. Context awareness is among the core components of these research challenges.

Our conceptual framework that shows the basic elements of a pervasive computing environment is given in Fig. 1. It consists of pervasive environment, context modeling and collaborative context-aware service.

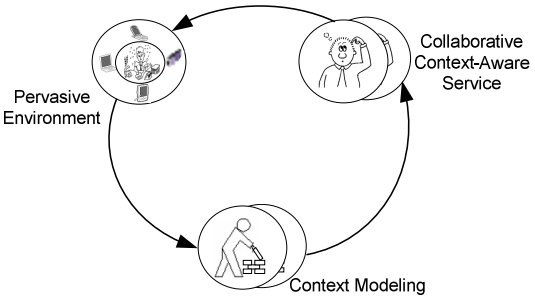

Fig. 1 Conceptual framework for pervasive context-awareness

Pervasive environment is characterized by dynamicity, heterogeneity and ubiquity of users, devices and resources, ad-hoc connection among the devices and existence of hardware and software sensors. It comprises plenty of interacting gadgets most of them tiny and sometimes invisible. Users in such environment are sometimes frustrated by the gadgets surrounding them. Among the challenges are to know when and under what situation we operate these gadgets and the time allocated for such activity.

HCoM modeling deals with how context data is collected, organized, represented, stored and presented. CoCA service interprets and aggregates the low level context values to a more meaningful high level context like

Based on "CoCA: A Collaborative Context-Aware Service Platform for Pervasive Computing", by Dejene Ejigu, Marian Scuturici, Lionel Brunie, which appeared in the Proceedings of the IEEE International Conference on Information Technology: New Generations (ITNG'07), Las Vegas, USA, © 2007 IEEE.

changing geographic coordinate to street names. It then performs reasoning about the context and passes decisions about the actions to be triggered. It stores knowledge and decisions corresponding to the context instance into knowledge repository for future reference. Most computing devices in pervasive computing environment are tiny and are resource hungry. On the other hand, acquiring, storing, reasoning, and processing of context information to provide context-aware service requires a considerable amount of hardware resources. This calls for collaboration and combined efforts of these devices.

The objective in this work is therefore to propose and investigate architecture for context-aware services focusing on context reasoning in pervasive computing environment based on hybrid, semantic and collaborative approaches.

The rest of the paper is organized into the following sections. In section II, we discuss related works. Section III is about the hybrid context model used in the development of the platform. Section IV presents our innovative context-aware service platform. Section V discusses the collaborative aspect of the platform. In section VI, we present implementation and usage scenario of the platform and finally in section VII, we give conclusions and highlights of future works.

II. Related Works

Several projects have introduced context-aware computing as a key feature in their application development over the last decade and many works have been done so far that demonstrate the importance of context awareness in pervasive computing [1]. Earlier efforts focused on the development of application specific context-aware systems. Such systems exist in the form of Cyber Guides [2], Cool Town [3], Cricket Compass [4], etc.. They use only few types of context information mainly identity, time and location.

Oxygen[5] and Gaia[6] projects are among the major contributions towards achieving context awareness services. They are however limited to infrastructure based environment that does not allow ad-hoc communications, one of the important features of pervasive computing environment.

CoBrA-ONT [7] aims to develop architecture to enable distributed agents to control the access of their personal information in a context-aware environment. It uses Semantic Web Technology. CONON [8] is a context Ontology for reasoning and representation of contexts in pervasive environments. Their approach is based on the treatment of high-level implicit contexts that are derived from low-level explicit contexts. Gu et al. [9] have proposed a service based framework and middleware solution that uses context reasoning schemes as a basic source of context-aware services. MARKS[10] and RCSM[11] have partially addressed and have emphasized the importance of ad-hoc communication and knowledge usability in the development of context-aware systems in pervasive computing environment. But the use of semantic ontology are sources of domain knowledge and ad-hoc sharing of this knowledge among pervasive devices with varies capacity is a key issue in a pervasive context-aware computing that is not well addressed by the existing works.

There is still a lot to do to ensure standard, scalability, quality of service and robustness with respect to the inherent pervasive component failure that needs self healing, varying device capacity, and frequent change of context in a pervasive environment. The computationally intensive characteristics of context reasoning process and lack of semantically rich context model have also been a bottleneck for the development of pervasive applications.

Our work proposes a neighborhood based collaborative context-aware service platform that facilitates reusability of context resources and reasoning axioms, and sharing of computational resources among devices in the neighborhood space using semantic ontology based hybrid model as a core data source.

III. Semantic Based Hybrid Context Model

Context can be described in terms of a statement that we make about the characteristics of a context entity. Some examples of how we define context data can be:

Generic level	**Domain level equivalent**
Person isEngagedIn Activity	Physician isEngageIn Patient-treatment
Person isLocatedIn Location	Student isLocatedIn Library

A close look at such statements shows the need for a higher level statement about these statements that can be expressed in terms of meta-information or axioms. Information like time, precision and source of context data can be part of such meta-information. For example, if we state *located-in* is *transitive* then: "*library is located in campus"* and "*student is located in library"* means that "*student is also located in campus"*. Similarly, if we say *owned-by* is an *inverse of owns* then: "*device is owned-by person"* means "*person owns device"*. Another example is that we can state *"student is located in library"* at a given *time t*. We can also state that the precision of the statement *"a network connection has low speed"* is 85%. Here, for example, *time t* that is associated with "*student is located in library,"* refers not to the individual components of the statement but to the entire statement.

A. Context Representation

Context representation needs a model that supports easy transaction of piece wise tiny but voluminous, and dynamic context data. Equally important is the capacity to aggregate and interpret the context data with respect to its semantics so as to make it ready for reasoning and decision. We have developed EHRAM, a context model based on its sources

that consist of set of hierarchies (H) of set of entities (E), set of entity relations (Re), set of attribute relations (Ra), set of axioms (A) and set of metadata (M). We name this structure the EHRAM layered context representation structure (or in short the EHRAM model).

Consider an application in a medical domain where context data comes from medical entities like patients, doctors, nurses, activities and events in the hospital, devices, locations, etc. Example of Graphical representation of EHRAM model with data from the medical domain is given in Fig. 2. The domain layer (lower block) of the graph shows entities, concepts, relations, axioms and metadata drawn from medical domain. Entities like *activity, person, and device* in the generic layer (upper block) are common to all domains of applications. They are high level context entities from which sub context entities can be derived. Relations like *isEngagedIn(Person, Activity), owns(Person, Device), isLocatedWith(Person, Person), isLocatedWith(Person, Device), isLocatedWith(Device, Person) and hasMemory(Device, memory)* represent generic relations that can be inherited down in the hierarchy by the sub-entities and instances in the specific domain of application.

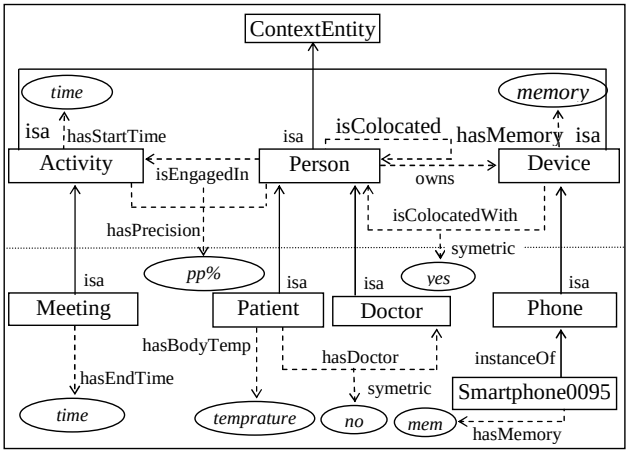

Fig. 2 Example of EHRAM model representation

Entities like *patient, doctor, meeting and phone* in the domain layer are specific to the medical domain of application. In addition to the relations that can be inherited from upper layer in the hierarchy, we can also define relations that belong to a specific entity in the domain of application and its derivatives in the lower hierarchy. Such inheritance and addition of new relations to new entities or instance continues in the same fashion down in the hierarchy. Among such relations defined in the medical domain layer example are: *hasBodyTemp(Patient, tmp), hasDoctor(Patient,Doctor), hasEndTime(Meeting, time).*

The primary characteristic of a context data is that it possesses an actor or a *subject*. The type and value of the context is expressed in terms of multiple properties. In our discussion, we use the terms *predicate* and *object* to represent the situation of the subject with respect to a specific property. This convention goes with the RDF-triple naming style which we intend to use for modeling context using Ontology. *<subject predicate object>* are, therefore, the basic triples to represent context. Additional metadata about the basic context triple can also be included as part of the context data. In addition to the subject, predicate and object triples, context modeling requires context attributes like source, time, place, validity, claims, doubts, proofs, etc. to describe the triple itself. This can be demonstrated using RDF reification where each original statement serves as a resource for the other additional statements made about it. We can demonstrate this using: *"Bob is located in the Library"*. This statement can be reified by additional meta-information like *"is reported by sensor #5", "has accuracy of 88%"*, etc.

```
//Reification
<ns:Bob ns:isLocatedIn ns:Library/> //original statement
<ns:XX rdf:type resource=rdf:Statement/> //reification starts
        <ns:XX rdf:subject resource= ns:Bob/>
        <ns:XX rdf:predicate resource=ns:isLocatedIn/>
        <ns:XX rdf:object resource= ns:Library/> //reif. ends
<ns:XX ns:isReportedBy "Sensor#5"/> // meta-context on XX
<ns:XX ns:hasAccuracyOf "88%"/>// meta context on XX
```

In order to extend our EHRAM context representation model to a more generalized context management model, efficient storage and retrieval, ease of serialization and semantic support are necessary. In the EHRAM model, we have parts that represent the semantics of the data (context knowledge) and parts that represent the context data. For proper reasoning and decision in a context-aware computing environment, both context knowledge and context data should be treated equally in the process of developing a context management model.

Based on our ER to EHRAM model mapping algorithm and works like [20], we can represent the data part of context into a relational database. Similarly, the semantic part of context data can be represented using semantic ontology like in our previous work [12]. In relational approach, we can map our EHRAM model to relational tables. Relational models provide standard interfaces and query optimization tools for managing large and distributed context database or receive and send notifications on context changes. In the ontology approach, we can create direct mapping of concepts in the EHRAM model to semantic ontology representation using owl and rdf schema. Ontology representation tools provide a widely accepted and formal representation of context knowledge in order to interpret and reason about context information.

B. Hybrid Approach to Context Management

Both relational and ontology based approaches when used for context management modeling, have their own pros and cons. Ontology tools are only good at statically representing the knowledge of a domain. They are not designed for capturing and processing constantly changing information in dynamic environments in a scalable manner.

Moreover, existing ontology languages and serialization formats are text based (xml/rdf/owl) and therefore are not designed for efficient query optimization, processing and retrieval of large context data. Similarly, we can not fully manage context data using standard database management principles because in database management systems, like in relational database, the meaning of the data is in the "head" of the user.

Our rational behind the need for the hybrid context model, therefore, is to separate context data management and context knowledge management, process them separately and put the results together for better reasoning and decision support in a context-aware environment. We use ontology approach to manage context semantics and relational approach to manage context data. We name this combination a Hybrid Context Management (HCoM) model. HCoM model aims to combine the bests from the two worlds. It is an upgrade on our earlier ontology based generic context management model, GCoM [12], [16], [18].

HCoM has a heuristic component. Heuristics is the application of experience-derived knowledge to a problem-solving process where as the adjective heuristic refers to using the problem-solving technique in which the most appropriate solution of several alternatives is selected at successive stages [22]. The heuristic component of HCoM provides a means to select and load only part of the large static context data that is accumulated over a period of time depending on who and where the user is, the intended activity to which the user is going to be engaged, devices available for use, institutional policies etc. It uses matching patterns gained through experience to identify relevant group of context data. Through analysis of history profiles, heuristic filtering assigns a numerical score to each class of entity in relation to a particular request instance. This score is used to determine whether the context entity class is relevant or not.

Loading only relevant data for reasoning minimizes the size of the reasoning space and reduces the unnecessary overloading of the reasoner so as to improve the overall performance. It helps to overcome limitations of lack of scalability of most of the reasoning systems to the ever increasing volume of reasoning resources in the pervasive environment.

Fig. 3 shows components and functionalities that represent HCoM model. The arrows represent flow of different set of data from or to each component: (a) ontology data, (b) context data, (c) requirement and preference for heuristic selection, (d) static context data, (e) context data populated into the ontology during runtime, (f) context event notice, (g) query and retrieval of relevant static context data during initialization, (h) relevant static context data populated into the ontology, (i) aggregated context data and ontology, (j) context data model, (k) rules into the context model, (l) transaction for collaboration with neighborhood, (m) transaction for collaboration with neighborhood.

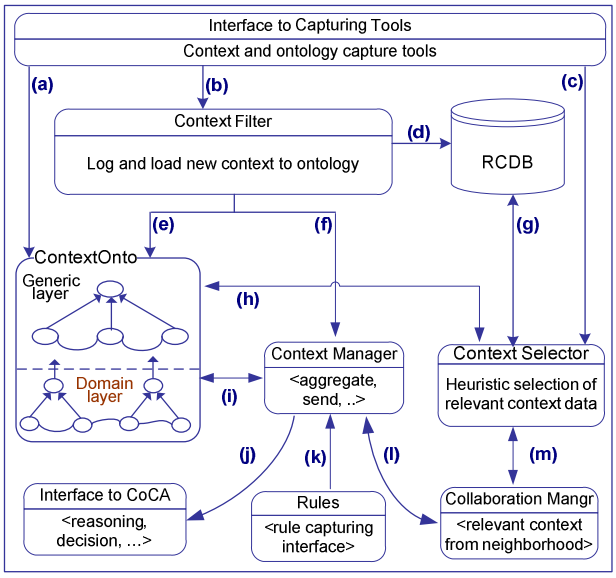

Fig. 3 Architecture of HCoM model

IV. Enhanced CoCA Service Platform

A context-aware service in pervasive computing aims at acquiring and utilizing context information to provide appropriate services. For example, a cell phone is always set to vibrating mode when its holder is in the library if it has the knowledge about his current location. The reasoning engine in the platform accepts a set of context data, rules and their semantics and changes it into concrete knowledge necessary for reasoning and decisions. The platform should be independent of the data elements.

A. Overview

In this section, we present an enhanced CoCA, a neighborhood based hybrid and Collaborative Context-Aware service platform. Enhanced CoCA is based on our previous work on CoCA [18]. Fig. 4 shows architecture of the CoCA service platform classified into four functional groups: interface, data source, core service and supplementary service.

The *Interface Manager* controls a user interface and an interface between the CoCA platform and other modules specific to domains of applications. It also hosts action triggering process depending on the specific application domain in which the platform is used. The *Data Source* in the enhanced version of CoCA has a heuristic component with hybrid features that helps to handle context selection and loading from relational data store to the semantic schema to facilitate speedy reasoning. The *Core Service* provides the context-aware service after performing reasoning and decisions. The *Supplementary Service* on the other hand consists of services like knowledge discovery service, collaboration service, security service, etc. Knowledge discovery service, for example, adds features to enhance learning capacity of the CoCA platform.

Collaboration service adds features for collaboration between neighbors in the CoCA neighborhood space.

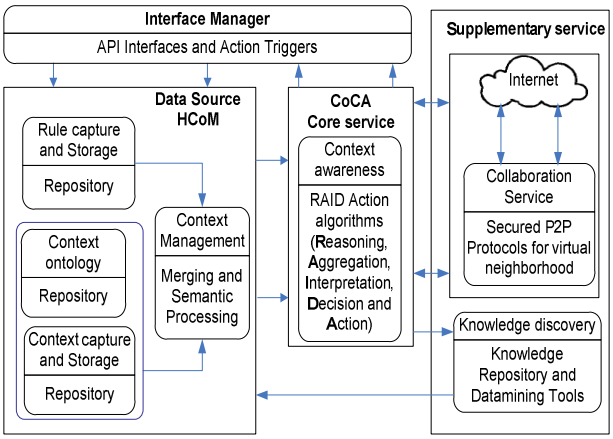

Fig. 4 Architecture for the CoCA platform

B. CoCA Core Service

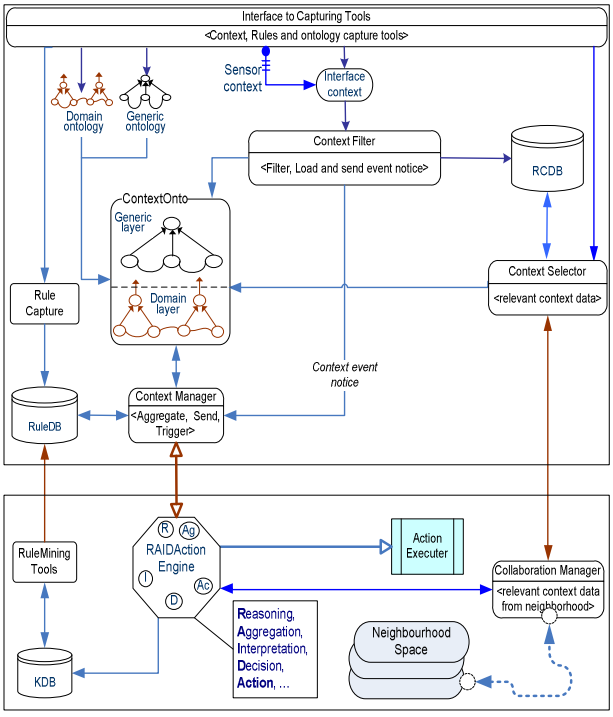

Fig. 5 Component view of the Enhanced CoCA platform

Core service uses input data from the HCoM data model and provides the necessary context-aware service after reasoning on the components. It consists of the *RAID-Action engine* (Reasoning, Aggregation, Interpretation, Decision and Action engine) that performs reasoning and decision about the actions to be triggered. It is responsible to interpret the captured low level context to high level meaningful context. Coordinate based location values, for example, are interpreted to street names. It also performs aggregation by combining two or more low level contexts to one meaningful high level context. Aggregation of body temperature, heart rate and blood pressure of a patient can be used to tell patient's health condition (context). Reasoners play an important role in the RAID-Action process. They derive conclusions from collections of context data, rules and the ontology.

Component view of the enhanced CoCA platform with new functionalities like context selector is given in Fig. 5. The RAID Action Engine being at the heart of the service platform uses contexts, rules and ontology combined as an input. Each has two separate sources; context data are either from the user interface or from sensors, rules are from the user interface or from the rule-mining module. Ontology also has two sources; generic ontology that serves multiple domain of application and domain dependant ontology.

```
//Relevant reasoning data model for a Person
HeuristicSelector=new HeuristicSelectorClasss()
spLive = HeuristicSelector.getLiveProfile(Person)
spHistory = HeuristicSelector.getHistoryProfile(Person)
selectionProfile = HeuristicSelector.setSelectionProfile(spLive, spHistory)
relevantTriples = HeuristicSelector.selectRelevantTriple(selectionProfile)
if (HeuristicSelector.inSufficient(relevantTriples))
{
  rTriples = HeuristicSelector.callDiscoverInNeighborhood(Person)
  relevantTriples = HeuristicSelector.addRelevantTriple(relevantTriples, rTriples)
}
HeuristicSelector.loadTriplesToOntology(relevantTriples)
```

Fig. 6 Heuristic context selector algorithms

One of the major contributions in the enhanced CoCA is the selective feature of loading only relevant context data into the reasoner. Loading only relevant data minimizes the reasoning search space and reduces the unnecessary overloading of the reasoner so as to improve the overall efficiency of the service. It uses heuristics methods on user information, devices available, institutional policies etc. to select and load only part of the context data. It helps to overcome limitations of lack of scalability of most reasoning systems. Steps involved in heuristic selector algorithm are given in Fig. 6.

C. CoCA Action Triggers

Actions are major outputs of the CoCA services. Algorithms for action triggering in CoCA are based on multifaceted approach [21].

Multifaceted action or adaptation processing coordinates action triggers based on both sets of decisions from CoCA-RAID and some other factors such as priority or existence of some other actions currently triggered. It is based on the idea that an application consists of both components that are context sensitive and components that do not depend on the context, and a context sensitive component can be seen as an item with many facets. In CoCA platform, a facet has

a condition that behaves like a switch; in the sense that if the condition is true the facet is exposed otherwise the facet is hidden. The principle is demonstrated in Fig. 7.

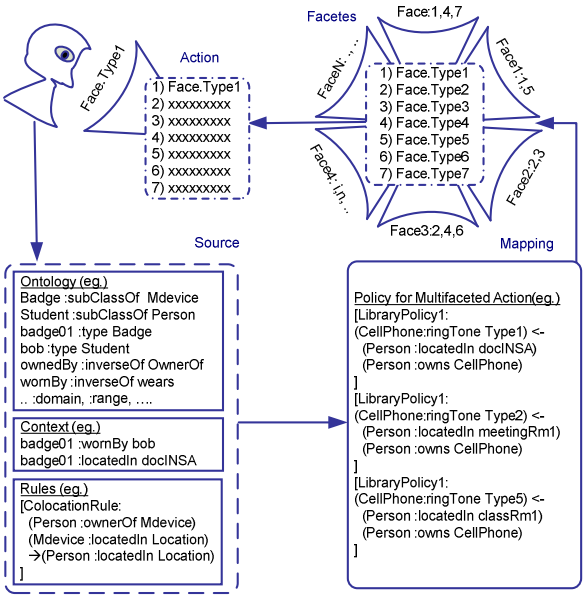

Fig. 7 Principles of multifaceted action processing in CoCA

V. COLLABORATION MANAGER

Collaboration Manager is a module that works based on peer-to-peer negotiation and communication protocols. The neighborhood space consists of computing devices with varying capacities. Each device in the neighborhood space is assumed to have a minimum configuration of the CoCA service platform. It may send decisions or decision support data back to the source or perform the action by itself depending on the context of the environment.

Storage and processing of context data to knowledge is highly resource intensive while on the other hand most ubiquitous devices in the pervasive world have scarce resources. Mobility and anytime/anywhere access requirement of pervasive users make the problem worse. To overcome this problem, we propose to investigate a collaborative approach where devices combine their resources. Capable devices, like standard PCs, play an important role of "big" brothers to support tiny devices like PDAs and smart phones. With the current trend of PC availability, we can assume that, in the neighborhood space, we always have capable devices that play a supporting role.

The role of collaboration manager in the CoCA service platform is to share computing resources like context, rules, ontology, processor, memory, etc. to solve computing problems to provide comprehensive context-aware service which would otherwise be difficult and some times impossible for a single pervasive device to solve. This module uses the principle of virtual network overlay indicated in Figure 8. Among the basic requirements for collaborative computing between CoCA peers in the neighborhood space is the ability to self-organize into peer groups, discover each other and each other's services and resources. JXTA [19] as a supporting technology is a set of open protocols that allow any connected device on the network ranging from cell phones and wireless PDAs to PCs, servers and super computers to communicate and collaborate in a peer-to-peer manner.

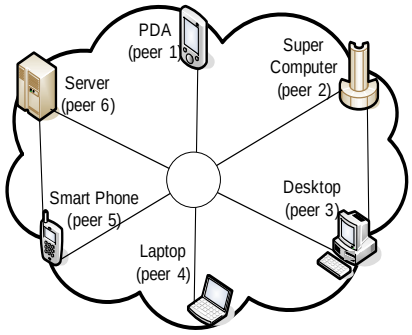

Fig. 8 Overlay of the virtual collaboration network

An example showing a code that uses Jxta protocols for resource advertisement and discovery in the CoCA service platform and the corresponding XML advertisement is shown in Fig. 9. Such location based context information in our experiment is obtained by location prediction module described in our earlier paper [17].

In a collaborative peer-to-peer environment like that of CoCA, we need to support different levels of security and resource access. Security between peers in the CoCA service environment can be achieved by the Jxta security protocol in which peers operate in a role-based trust model, in which an individual peer acts under the authority granted to it by another trusted peer to perform a particular task.

```
//Location detector–publishes LocationAdvertisement
    LocatioAdvertisement adv = new LocationAdvertisement();
    adv.setID(myID);
    adv.setName("PDA001");
    adv.setOwner("Bob");
    adv.setLocation("Library");
    discoveryService.publish(adv, DiscoveryService.ADV, lifeTime,expTime);

//Clients retrieve by discovering location advertisements
    discoveryService.getRemoteAdvertisements(null,
                DiscoveryService.ADV,"Name", "PDA001*");
    Enumeration advs = discoveryService.getLocalAdvertisements(
                    DiscoveryService.ADV, "Name", "PDA001*");
    LocationAdvertisement adv = null;
    while (advs.hasMoreElements()) {
            adv = (LocationAdvertisement) advs.nextElement();
            String ownedBy = adv.getOwner();
            String locatedIn = adv.getLocation();
            break;
  }

//Sample advertisement file that contains context data
<jxta:LocationAdvertisement xmlns:jxta="http://jxta.org">
  <ID> urn:jxta:uuid 59616261646162614A7874615… … </ID>
  <Name> PDA001 </Name>
  <Owner> Bob </Owner>
  <Location> Library </Location>
</jxta: LocationAdvertisement>
```

Fig. 9 Resource advertisement and discovery

VI. IMPLEMENTATION AND EVALUATION

An activity diagram showing details of invocation and flow of activities in the CoCA service platform is shown in Fig. 10. The two dimensional 2x2 partitions in the diagram show the services and the units responsible for each activity. The vertical partitions show the services responsible to do the activity (the context modeling service or the context-aware service) and the horizontal partitions show the units responsible to do the activity (the base unit itself or the neighborhood space).

Prototype of CoCA has been tested for scalability and multi-domain support by data from domains like application adaptation [16] in computing domain and a smart hospital in medical domain. A graphical representation of part of context ontology for the hospital scenario is given in Fig. 11.

In this section, we mainly discuss the internal functions of CoCA and its action engine RAID by experimenting on the prototype of our architecture and its data model, HCoM. Protégé [13] ontology development tools and Jena framework [14] are used to implement reasoning and decision algorithms.

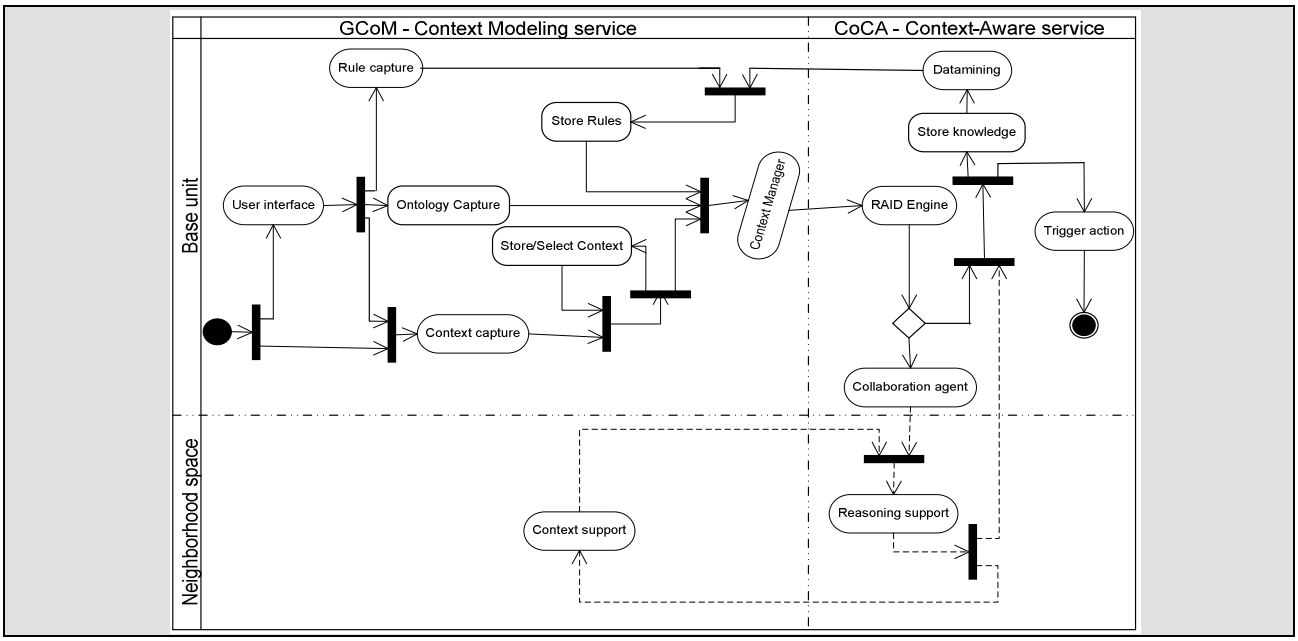

Fig. 10 Activity diagram for the CoCA service platform

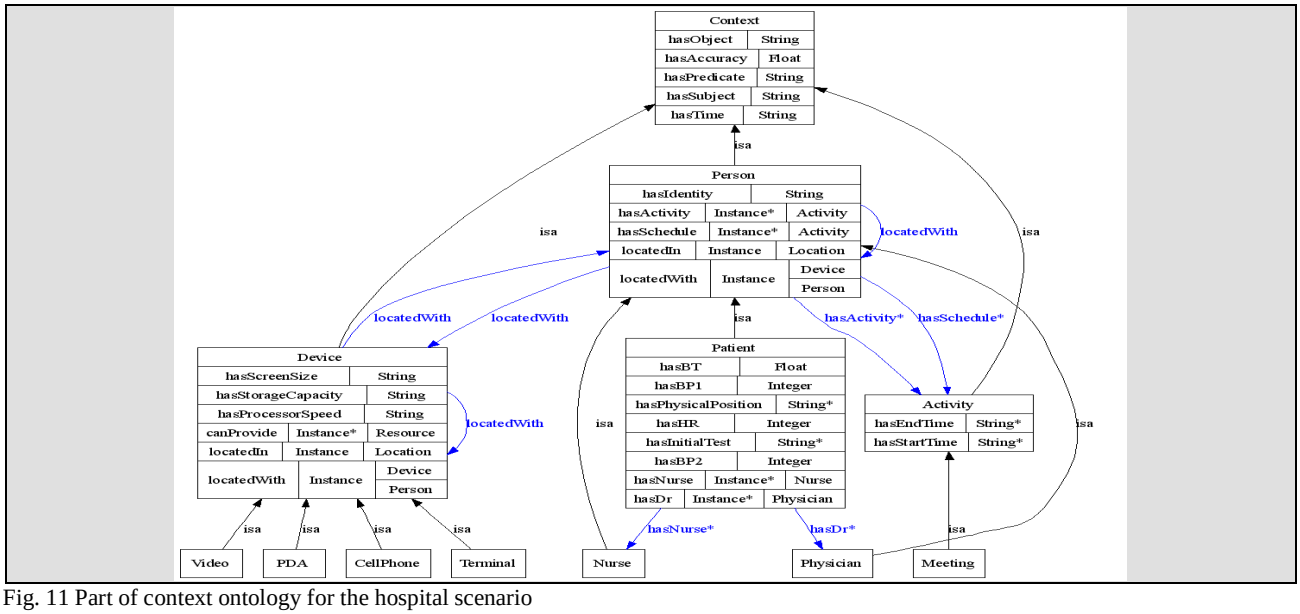

Fig. 11 Part of context ontology for the hospital scenario

A. The PiCASO Scenario

We use a pervasive campus scenario, *Pervasive Campus-Aware Smart Onlooker (PiCASO)*, to demonstrate the internal functions. PiCASO is based on the scenario of a university campus where research students and professors are involved. Besides the scheduled regular meetings among students and professors, informal and spontaneous meetings and discussions are important for the advancement of their work. Discussion can take place among two or more of the researchers depending on the relevance of their work.

The questions that can be raised in this scenario are: When do they make such a meeting? How can only those available are informed about someone else's interest to discuss about a specific subject matter during his tea break? How can a student know that his professor is available for the coming 30 minutes? How can a student know when his professor is in the tea room, in his office or in the corridor passing by the office of the student? What type of messaging method is appropriate to send such information to a particular person located at a particular place at a particular time? If telephone is used to accept such a message, what should its call mode be (vibrating, ringing)? And so on.

```
Ontology
<xml version="1.0"?>
<rdf:RDF .....
 <owl:Class rdf:ID="Student">
     <rdfs:subClassOf> <owl:Class rdf:ID="Person"/> </rdfs:subClassOf>
 </owl:Class>
 <owl:Class rdf:ID="Library">
     <rdfs:subClassOf> <owl:Class rdf:about="#Location"/> </rdfs:subClassOf>
 </owl:Class>
 <owl:ObjectProperty rdf:ID="ownedBy">
     <rdfs:range rdf:resource="#User"/>
     <rdfs:domain rdf:resource="#Device"/>
     <rdf:type rdf:resource="http://www.w3.org/.../owl#FunctionalProperty"/>
   <owl:inverseOf> <owl:ObjectProperty rdf:ID="ownerOf"/> </owl:inverseOf>
 </owl:ObjectProperty>
 <Student rdf:ID="Bob">
     <ownerOf>
         <PDA rdf:ID="PDA001">
           <hasScreenSize
         rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Medium
           </hasScreenSize>
         </PDA>
     </ownerOf>
     <ownerOf rdf:resource="#Cellphone001"/>
 </Student>
.....
</rdf:RDF>
```

Fig. 12 HCoM model ontology representation using owl

For the purpose of context data representation we use the ontology based generic context management, HCoM, model. The three important data sources related to the HCoM model are context ontology, context instance data and derivation rules. The RDF/OWL [15] version of part of the context ontology for the PiCASO is given in Fig. 12. Sensed context is to be communicated using XML/RDF-triple representation format as indicated in Fig. 13.

In ontology, we have semantics about all components and related concepts. It also defines relations between the components and the concepts. For example, a person can be defined as an *ownerOf* a device and then the relation *ownedBy* is automatically granted to the reverse relation because both concepts are defined, in the ontology, as being the inverse of one another. One example of such context instance data can be given as: *Bob is ownerOf PDA001*. This means, a student known as Bob is *ownerOf* a PDA device called PDA001, from which, the ontology reasoner can easily deduce that PDA001 is *ownedBy* Bob.

In the HCoM model persistent data about static contexts (e.g. ownership relationship of persons to devices like telephone or PDA) can be stored as profiled or static context in any standard database format which can then be selectively populated as context instances into the ontology structure at runtime.

```
//Context
<xml version="1.0"?>
<rdf:RDF ... ..
    <Professor rdf:ID="Dave">
          <locatedIn rdf:resource="#Room01"/>
          <engagedIn rdf:resource="#FormalMeeting"/>
    </Professor>
    <Student rdf:ID="Carol">
          <locatedIn rdf:resource="#ReadingRoom"/>
    </Student>
    <Student rdf:ID="Alice">
           <locatedIn rdf:resource="#Room03"/>
    </Student>
    <Professor rdf:ID="Eve">
          <locatedIn rdf:resource="#Room02"/>
          <engagedIn rdf:resource="#Browsing"/>
    </Professor>
    <PDA rdf:ID="PDABob">
    </PDA>
    <PC rdf:ID="PCAlice">
    </PC>
    <Messaging rdf:ID="MessengerService">
    </Messaging>
    ... ... .
</rdf:RDF>
```

Fig. 13 HCoM model context representation using RDF

Sample rules for representing explicit wishes using Jena generic rule format are given in Fig. 14. Such data are stored on a disk file that can be in any suitable format: text or any Jena compatible database format like MySQL. Our example in this scenario assumes the use of text files stored in a local disk file for context, ontology and rules data. After semantically combining together data from all the three sources, the reasoner can draw parameters for the actions in the PiCASO.

Action parameters are extracted from the reasoner model using queries. In this case, we use a SPARQL query language. SPARQL is suited for RDF triple and is supported by Jena based data models. Sample query to select all instances of context, relations and values from the model and demonstration of the reasoning process using a simple trace of a library rule is given in Fig. 15. It shows an output message with parameters (CellPhone001, setRingingTone, SilentMode) that are used in a call to a customized action trigger module. In this particular case, the action is to set the ringing mode of Bob's cell phone to silent mode.

```
//Rules
//Ontology based derived rules
[Transitive_Rule: (?a coca:locatedIn ?b)
                  (?b coca:locatedIn ?c)
                 ->(?a coca:locatedIn ?c)]
[Inverse_Rule: (?a coca:ownerOf ?b)
              ->(?b coca:ownedBy ?a)] //inverse
.....
// Domain based phone management rules
[locatedRule:(?device coca:locatedIn ?location)
    (?device coca:ownedBy ?person)
    -> (?person coca:locatedIn ?location)
]
[libraryRule:(?student coca:locatedIn coca:Library)
    (?student coca:owns ?phone)
    -> (?phone "setRingTone" "silent")
]
[classRule:(?student coca:hasSchedule ?class)
    (?class coca:isScheduledIn ?classRoom)
    (?class coca:startTime ?t1)
    (?class coca:endTime ?t2)
    ((?Student coca:locatedIn  ?classRoom) coca:hasTime ?t)
    (?t sys:greaterThan ?t1)(?t sys:lessThan ?t2)
    (?student coca:owns ?phone)
    → (?phone "switchMode" "Vibrating")
]
[meetingRule:(?student coca:hasSchedule ?meeting)
    (?meeting coca:scheduledIn ?meetingRoom)
    (?meeting coca:startTime ?t1)
    (?meeting coca:endTime ?t2)
    ((?student coca:locatedIn ?meetingRoom) coca:hasTime ?t)
    (?t sys:greaterThan ?t1) (?t coca:lessThan ?t2)
    (?student coca:owns ?phone)
    →(?phone "switchMode" "Silent")
]
xcampusRule:(?Student coca:locatedIn OutSideCampus)
    (?student coca:owns ?phone)
    → (?phone "switchMode" "MusicRingingTone")
]
```

Fig. 14 HCoM model related rule representation using Jena

```
//Sample SPARQL query
PREFIX rdfsyntax: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX xmlschema: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX coca: <http://www.cocasp.fr/forumf.owl#>
SELECT DISTINCT ?DomainSubClass  ?SS ?RR ?VV
WHERE {
        ?GenericClass rdfs:subClassOf coca:Context.
        ?DomainSubClass rdfs:subClassOf  ?GenericClass.
        ?SS rdf:type ?DomainSubClass.
        ?SS  ?RR  ?VV.
        FILTER( ?SS != ?VV).
        FILTER( ?RR != rdf:type).
        FILTER( ?RR != owl:differentFrom).
        FILTER( ?RR != owl:sameAs).
        FILTER(?GenericClass != ?DomainSubClass).
        FILTER(?GenericClass != coca:Context).
 }
 ORDER BY ?SS ?RR ?VV.

//Trace of reasoning process after query using libraryRules:
//Given sensed context
(PDA001 locatedIn ReadingRoom)
// transitive property of locatedIn defined in the ontology
    =>(PDA001 locatedIn Library)
// locatedRule in the derivation rule list and the fact
//(Bob owns PDA001) in the context list
    =>(Bob locatedIn Library)
// libraryRule in the derivation rule list and the fact
//(Bob owns CellPhone001) in the context list
    =>(CellPhone-01 SetRingTone silentMode)
```

Fig. 15 Sample query and trace of reasoning process

Fig. 16 shows a java code segment that uses Jena API to put together all the major components of the CoCA service platform for reasoning, inferences and decisions. It also indicates how SPARQL queries are used to draw parameters for action triggering. This simple example demonstrates the capacity of the CoCA platform to use our context model in building a context-aware service.

B. Implementation

```
//imports …
.. ..
public class CoCASupport{
public static void main(String[] args) {
    //Model Setup
    GenericRuleReasoner reasoner = new GenericRuleReasoner((List)
            Rule.rulesFromURL(    "file:ForumRules.rules"));
    OntModel tempModel = ModelFactory.createOntologyModel(
            OntModelSpec.OWL_MEM_MICRO_RULE_INF,
            ModelLoader.loadModel("file:ForumOntology.owl"));
    InfModel cocaModel=ModelFactory.createInfModel(
            reasoner,tempModel);   //Example usage
    String queryString = "PREFIX coca: <http://www.owl" +
        "-ontologies.com/unnamed.owl#> "+
        "SELECT ?phone WHERE {?phone coca:setRingTone coca:Silent.}";
    Query cocaQry = QueryFactory.create(queryString) ;
    QueryExecution qexec=QueryExecutionFactory.create(cocaQry,cocaModel) ;
    ResultSet cocaResults = qexec.execSelect() ;
    for ( ; cocaResults.hasNext() ; )
    {
        QuerySolution res = cocaResults.nextSolution() ;
        RDFNode phone = res.get("phone") ;
        System.out.println("Setting ringing tone of "+
                            phone +" to silent");
        fireAction("RingingTone", phone,"silent"); //Module Call
    }
    qexec.close();
 }
}.. ..
```

Fig. 16 Reasoning elements in the CoCA platform

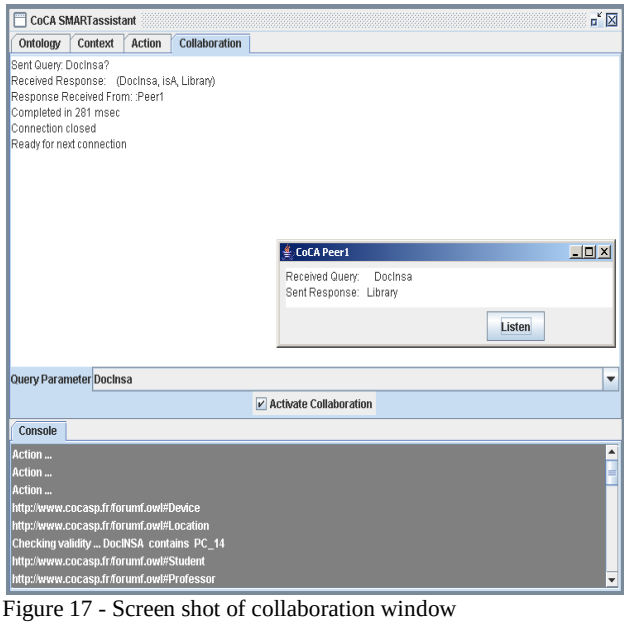

Figure 17 - Screen shot of collaboration window

PiCASO implementation consists of 2,025 static context instances handcrafted from 177 entity instances, 44 defined relations of which 28 are used. Such instances are stored in the RCDB. Relevant context data is then loaded into context-onto using heuristic filtering during initialization. It also consists of some 80 lines of rules created from domain policies and user needs. We have also used some 40 lines of dynamic context instances. Combining all these with the 500 lines of owl ontology schema, the PiCASO HCoM model produces a total of 2,412 triples in its entire reasoning space.

CoCA smart assistant has 4 major interactive screens: Ontology Screen, Context Screen, Action Screen and Communication Screen. A sample screen shot of the Communication screen is given in figure 17.

C. Evaluation

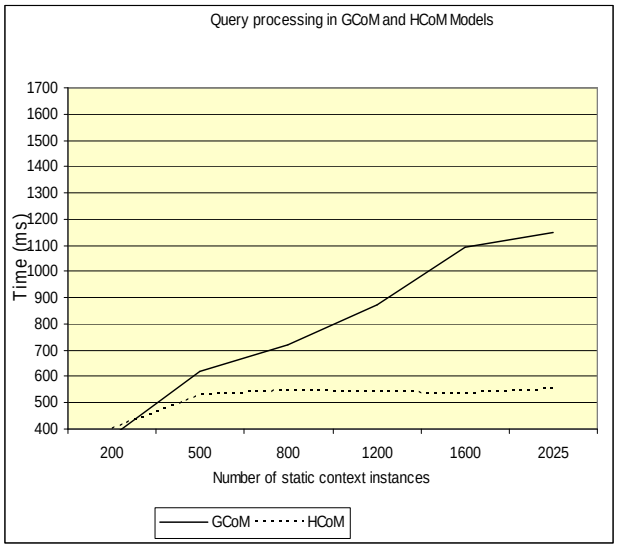

Figure 18 - Scalability Measures for GCoM and HCoM

To overcome the expensive time complexity problem, HCoM, a data model in enhanced CoCA, uses selective loading of reasoning resources. Only relevant context is loaded for reasoning at a time. In our experiment this gives a time-graph that tends to remain constant with the growth in the volume of context data. The use of our earlier generic data model, GCoM, on the other hand, produces a constantly growing time graph with the increase of volume of reasoning resource. Figure 18 shows a time trend graph for reasoning.

On the other hand, we have cases where wishes are not explicitly expressed using such type of rules and are called implicit wishes. In this particular scenario, for example, datamining tools can be used to learn the contexts under which the student switches the modes of his telephone. This means that the rules specified above are to be derived by the system itself. Such rules are dynamic in the sense that their quality improves with time. This is analogous to a "human assistant" assigned to help a student to change the phone ringing mode. If, for example, a human assistant is given orders like "now switch to X mode!", "now to Z!", "now again to X!", "now to Y!" etc. while moving around with the student, it is possible that the assistant can do some of these switching by himself after one week and with more accuracy after two or three weeks. Similarly, the datamining module is responsible to learn and propose users' wishes using the knowledge store. This is an autonomous decision and action support that helps the user not to be worried about the pervasive world routines.

Both rule guided explicit and autonomously learned implicit wishes result in proactive actions. As all input components, used in our platform (contexts, ontology and rules), are not hard coded into the system and are stored separately, it guarantees that the platform can be used in different domains by simply changing the inputs.

VII. CONCLUSIONS AND FUTURE WORK

Most of the current context management systems and context-aware services are based on ad-hoc models and domain dependent applications which may causes lack of desired formality, generality and expressiveness. Hence, management and use of context information and development of context-aware services in pervasive environments is still a challenge. Computing devices in pervasive computing environment on the other hand are tiny and have limited hardware resources. This calls for a domain independent collaborative context-aware platform.

In this work, we have presented an enhanced CoCA, a data independent and collaborative context-aware service platform that is based on hybrid context model. Initial prototype of the platform using a demonstration example on the smart campus scenario is implemented. A portion of our code for parsing and reasoning on the three components (ontology contexts and rules) and the SPARQL query is given as a demonstration. We also introduced a collaborative method to facilitate a peer-to-peer based neighborhood communication between the pervasive devices in order to share their resources. We have also tested this same module with data from a hospital scenario for application adaptation on patient-monitoring and follow up service [16] using the RAID-Action in CoCA platform.

Preliminary results from this implementation show that CoCA platform is a promising service that provide data independent environment for proactive context-aware computing services in pervasive environment.

More intelligent proactive services can also be achieved by enhancing the platform through its datamining component using data in the knowledge repository drawn from the RAID-Action engine of the CoCA platform. Therefore, as a continuation to this work, we are aiming to develop a complete prototype of small device deployable CoCA platform along with its hybrid context model. We will also continue to work on some evaluation benchmark, security and privacy issues.

VI. REFERENCES

[1] Mattern F., Sturm P., "From Distributed Systems to Ubiquitous Computing –State of the Art", Trends&Prospects of Future Networked Systems, pp. 3-25, Springer-Verlag 2003.
[2] Abowd G., Atkeson, Hong C.,J., Long S., Kooper R., Pinkerton M. "Cyberguide: A Mobile Context-Aware Tour Guide", Wireless Networks, 3:421--433, 1997.
[3] Kindberg T., Barton J. "A web-based nomadic computing system", Computer Networks, 35(4):443–456, 2001.
[4] Priyantha N., Miu A., Balakrishnan H., Teller S., "The Cricket Compass for Context-Aware Applications", Proc. of 7th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 2001), Rome, Italy, July 2001.
[5] Dertouzos M., "The Oxygen Project," Scientific American, Vol. 281(2), Aug 99, pp. 52-63.
[6] Cerqueira R., Hess C. K., Roman M., and. Campbell, R. H. "Gaia: A Development Infrastructure for Active Spaces", Workshop on Application Models and Programming Tools for Ubiquitous Computing in conjunction with UBICOMP 01, Sep 01.
[7] Chen H., Finin T., Joshi A. "An ontology for context-aware pervasive computing environments", Special Issue on Ontologies for Distributed Systems, Knowledge Engineering Review, Acapulco MX, August 2003.
[8] Wang X., Zhang D. Q., Gu T., Pung H. K. "Ontology Based Context Modeling and Reasoning using OWL", workshop on context modeling and reasoning at IEEE International Conference on Pervasive Computing and Communication , Orlando, Florida, March 2004.
[9] Gu T., Pung H. K., Zhang D. Q. "A Service-Oriented Middleware for Building Context-Aware Services", Journal of Network and Computer Applications, 28, 2005, pp. 1–18.
[10] Sharmin M., Ahmed S., Ahamed S.I., "MARKS (Middleware Adaptability for Resource Discovery, Knowledge Usability and Self-healing) for Mobile Devices of Pervasive Computing Environments," Third International Conference on Information Technology: New Generations (ITNG'06), pp. 306- 313, April 2006.
[11] Yau S. S., Karim F., Wang Y., Wang B., and Gupta S. K. S., "Reconfigurable Context-Sensitive Middleware for Pervasive Computing", IEEE Pervasive Computing, Vol. 1, No. 3, July-September 2002, pp. 33-40, IEEE Computer Society Press, Los Alamitos, USA.
[12] Ejigu D., Scuturici M., and Brunie L., "An Ontology-Based Approach to Context Modeling and Reasoning in Pervasive Computing", CoMoRea Workshop of the IEEE International Conference (PerCom'07), New York, USA, (to appear).
[13] Protégé from stanford.edu, Ontology editor and knowledge-base frame, http://protege.stanford.edu/ in December 2006.
[14] Jena from SourceForge.Net, A Semantic Web Framework for Java, http://jena.sourceforge.net/ in December 2006.
[15] McGuinness D. L., Van-Harmelen F., "OWL Web Ontology Language Overview", Available at http://www.w3.org/TR/owl-features/ in December 2006.
[16] Chaari T. , Ejigu D., Laforest F., Scuturici M. "Modeling and Using Context in Adapting Applications to Pervasive Environments", In the Proceedings of the IEEE International Conference on Pervasive Services (ICPS'06), Pages 111-120, Lyon, France, June 2006.
[17] Scuturici M., Ejigu D., "Positioning Support in Pervasive environments", In the Proceedings of the IEEE International Conference on Pervasive Services (ICPS'06), Pages 19-26, Lyon, France, June 2006.
[18] Ejigu, D., Scuturici, M., Brunie L.: CoCA: A Collaborative Context-Aware Service Platform for Pervasive Computing, Fourth IEEE International Conference on Information Technology: New Generations, ITNG'07, pp. 297-302, Las Vegas, USA, April 2007.
[19] JXTA, from JXTA project, http://www.jxta.org/, in February 2007.
[20] Roussos Y., Stavrakas Y., Pavlaki V.: Towards a Context-Aware Relatioal Model. In Proceedings of the CRR'05 Workshop, Paris, France, Jul 5-8, 2005.
[21] Anca Rarau, Kalman Pusztai, and Ioan Salomie, "MultiFacet Item Based Context-Aware Applications," International Journal of Computing & Information Sciences, Vol. 3, No. 2, 2005, pp.10-18, 2005.
[22] Paritosh, P.K. The Heuristic Reasoning Manifesto. In the Proceedings of the 20th International Workshop on Qualitative Reasoning, Hanover, 2006.

Dejene Ejigu is a Ph.D. candidate at LIRIS laboratory, INSA de LYON. He received his M. Sc. degree in Computer Science from the University of Wales Swansea, UK in 1989. In the past, until he started his Ph.D. study in 2004, he worked as a lecturer and researcher at the department of Computer Science, Addis Ababa University, Ethiopia. His research interests include pervasive systems, distributed computing and intelligent data management systems. His current research work focuses on context modeling and management, development of reusable context-aware application platforms, and their deployment in pervasive systems. More information can be found at http://liris.cnrs.fr/~edejene.

Vasile-Marian Scuturici (Ph.D.) is associate professor at INSA of Lyon since 2004. He received his PhD in computer science in 2001 at Lyon-2 University. His research interests include pervasive systems, video-on-demand systems, multimedia mining. His current work concerns the management of multimedia data in pervasive environments. More information can be found at http://liris.cnrs.fr/vasile-marian.scuturici.

Lionel Brunie (Ph.D.) is a full professor and director of the doctoral school of computer science of Lyon (EDIIS - Lyon). After he received his PhD in computer science at the Joseph Fourier University, Grenoble, Lionel Brunie joined Ecole Normale Supérieure (LIP lab) of Lyon as assistant professor. Then he took a University Professor position in computer science at the National Institute of Applied Sciences (INSA) of Lyon in October 1998 where he co-founded the LIRIS laboratory in 2002. Lionel Brunie leads a research team of 15 researchers and coordinates the Health informatics research action 40+ researchers). His main topics of interest include: collaborative multimedia information systems, multimedia databases, distributed systems, medical informatics. Lionel Brunie is the (co-)author of over 100 research papers; he has been member of over 30 scientific conference and workshop committees. More information can be found at: http://liris.cnrs.fr/lionel.brunie.