Journal of Network and Innovative Computing
ISSN 2160-2174 Volume 3 (2015) pp. 068-077
© MIR Labs, www.mirlabs.net/jnic/index.html

Development of Ontology based Middleware for Context Awareness in Ambient Intelligence

A.B. Karthick Anand Babu
Research Scholar, Department of Computer Science,
A.V.V.M. Sri Pushpam College (Autonomous), Poondi,
Thanjavur, Tamilnadu, India.
karthickanandbabu.ab@gmail.com

R. Sivakumar
Associate Professor, Department of Computer Science,
A.V.V.M. Sri Pushpam College (Autonomous), Poondi,
Thanjavur, Tamilnadu, India.
rskumar.avvmspc@gmail.com

Abstract—**There is currently lot of work in Ambient Intelligence particularly in context awareness. Context awareness enables service discovery and adaptation of computing devices for Ambient Intelligence application. In the same time, there is a common agreement of the fact that context aware systems should be responsive to Multi agents, assisting a large number of people, covering a large number of devices, and serving a large number of purposes. In an attempt to achieve such context aware systems with scalable scenario implementations, we propose an adaptive and autonomous context aware middleware using Ontology. Formal expressiveness and reasoning characteristics of Ontology make this middleware supportive to divergent programming applications. Our model provides a meta-model for context description that includes context collection, context processing and applications reactions to significant context changes. The advantage of the proposed ontology based middleware architecture is to improve the context awareness ability of the system and support divergent applications.**

Keywords: *Ambient Intelligence, Ontology, Context awareness, Reasoning, Multi-Agents*

I. Introduction

The term Ubiquitous Computing, introduced by Weiser [12], refers to the seamless integration of devices into users' in everyday life. This term represents an emerging trend towards environments composed of numerous computing devices that are typically mobile or embedded and that are connected to a network infrastructure composed of a wired core and wireless edges. The recent advances in Ubiquitous Computing, Ubiquitous Communication and Intelligent User Interfaces make Ambient Intelligence more dynamic and proactive vision of the futuristic computing. In present computing environments, systems should understand dynamic context and take consequent action. Context awareness enables service discovery and adaptation for ubiquitous computing devices. The term "context awareness" was first explicitly introduced in the research area of pervasive computing in it refers to the ability of computing systems to acquire and reason about the context information and adapts the corresponding applications accordingly. Context is a key issue for many research communities like ambient intelligence, real time systems or mobile computing, because it relates information processing and communication to aspects of the situations in which such processes occur. A context-aware system should automatically recognize the situation using various sensors. Therefore different user devices need semantically rich descriptive context models to provide shared understanding. Therefore, context modelling is an important feature in context aware systems. Therefore, to satisfy these necessities, it is necessary to separate the context awareness and application development, to separate the underlying sensing and context aware applications, a intermediate layer, particularly context-aware middleware is proposed with ontology. Ontologies facilitate information exchange and integration. Ontologies are used by web services so that the web can provide semantic web services to the humans. Ontologies may be specified using RDF syntax. Ontologies, as explicit formal specifications of the terms in a domain and the relations among them Gruber, 1993, are a widely accepted tool for modelling context information in pervasive computing domain. The reason for this acceptance is: (a) it has several advantages over other traditional modelling approaches and (b) the Semantic Web languages and tools have clearly gained maturity over the past years. Moreover, handling user query is another area where semantic knowledge is necessary. To this end, this work explores the benefits of ontologies to model the knowledge in a domain. This model aims to provide a machine readable unified format for abstraction of knowledge from divergent sensor data and to provide context aware services to the end user.

The proposed system, is a layered architecture with loose coupling and tight cohesion. In this model, each layer is designed with a specific role, which acts as input or output to the subsequent layer of the architecture. The feasibility of the presented work is validated and verified in the domain of educational applications.

Section 2 discusses related works for the proposed middleware; Section 3 represents problem statement and section 4 describes the main objectives of the proposed work. Section 5 discusses the architecture of our middleware. Section 6 explains the proposed middleware with a case study to demonstrate a context aware application. Section 7 concludes the paper with future discussion.

II. Related work

This section explains the related work in the field of Context awareness and Ontology based domain modelling for the system development.

In AmI, context-awareness specifies a nature of a context as temporal, spatial, personal, social and environmental. Ontologies represents concepts and associatations for a domain.

Eleni al. [15] present an ontology-based context modelling, reasoning process and management developed for composing context-aware Ubi Comp applications from Ambient Intelligence (*AmI)* artefacts. Laura Maria [13] proposes support for the design and implementation of a controlling service for context-aware applications by using Jess, a tool for developing rule-based systems. Georgalas [14] proposed ontology based reusable context model. This model facilitates the context reasoning by providing structure for contexts, rules and their semantics ontology. Wonil et al. [16] introduces Multi-layered Context Ontology Framework (MLCOF) based on rules for comprehensive, integrated context modelling and reasoning for object recognition. SOCAM [11] (Service Oriented Context-Aware Middleware) is a middleware for context-awareness that has an infrastructure for making context-aware applications. SOCAM has conjointly enforced a context reasoning engine that reasons over the knowledge domain. SOCAM discharges developers of context-aware services from the task of context management by providing a model to explain the context. However, there is no manner for developers to receive data from the sensor and this system specifies developers to implement adaptation service.

Rover Context Model Ontology (RoCoMO) [18] for modeling context for a given situation in pervasive computing environment . Hongyuan et.al. present a rule-based context-aware adaptation model. Although a rich landscape in adaptation related researches, a complete and generic context-aware adaptation approach is still missing. Several surveys dealt with service composition. Many of them classified the middleware under exclusive criteria such as manual versus automated, static versus dynamic, and so on. Lagares et al. presents, a context-aware ruled-based recommender system called as RING. The automatic redirection based on Semantic Web is proposed in RING. RING model permit users to receive required any kind of information through the best channel available depending on his personal preferences and context but, they do not concentrate on exact service to user context. In literature, many researchers argues the need to facilitate adapting context-aware services in the domain of ambient intelligence. As suggested by the chen et al.,[10] following are the reason for choosing ontology in the middleware to represent and reason on the context information:

i) knowledge sharing and dynamism
ii) well-defined declarative semantics
iii) efficient reasoning mechanisms
iv) interoperability support

Bei Wang et al. build up a ontology-based mobile information management model for the context information which lacks adaptation. Our model aims to overcome the demerits in the existing model and also to make use of the merits of ontology in middleware.

III. Problem statement

In context aware systems, meeting user requirements involves a thorough understanding of their interests expressed explicitly through search engine queries or implicitly through browsing behaviour and search context. User is not always able to describe the suited services due to the dynamic change of his context. This make as to design a system that is capable of providing user context aware service to the user in Ambient intelligence environment. Our design of the context attempts to address challenging issues such as developing explicit ontology representations of contexts, supporting context reasoning and maintenance through logic inferences.

IV. Objectives

The focus of our research is to design an ontology based middleware for context-awareness in Ambient Intelligent environment. In context aware systems, meeting user requirements involves a thorough understanding of their interests expressed explicitly through search engine queries or implicitly through browsing behaviour and search context. We consider that the user is not always able to describe the suited services due to the dynamic change of his context. To test and evaluate this context-aware functionality, laboratory exercise for divergent students is taken into consideration and relevant context is provided to students and their understanding to the subject is recorded. This proposed design will incorporate supporting intelligence to facilitate reasoning relating to the target applications. This paper presents ontology-based approach for developing context-aware middleware for the following objectives: (1) A well-defined, unified ontology enables knowledge sharing and reuse. (2) Ontologies with declarative semantics provide multiple policies to support context inference. (3) Ontologies provide various complex efficient inference mechanisms to deduce high-level contexts from low-level, raw context data, and to check inconsistent contextual information due to imperfect sensing. (4) Explicitly represented ontologies enhance the development of context-aware systems with semantic web technologies.

In addition to the above said objective, this paper aims to contribute designers and practioners to make a scalable architecture to build a simple context aware application for Ambient intelligence

V. Architecture

The proposed context-aware middleware architecture, showed in Fig.1, provides an ontology-based meta model for creating context aware applications.

Flow of Architecture

Step1 : Collect sensor data from physical and virtual sensors as xml form file.

Step 2 : Parse input and feed to the Context Acquisition &Fusion.

Step3 : Assimilate the context and input to context interpretation phase

Step 4&5 : Look up the Context Database and repository for such situation has occurred previously and retrieve relevant data according to the context and return to context interpretation

Step6 : Look up the Inference Rules for rules which takes premises existed in the context and returns a conclusion, if it is newly sensed data; learn it and update newly learned data in the database

Step 7 : Context Reasoning Engine query's a request to context representation.

Step 8 : Response to Context Reasoning Engine.

Step9 : Context Engine to perform the action and inputs to API/ Agent.

Step10 : Agent/API sends inputs to the Service server side and generates the alerts/warning signals to the Interface.

Step11 : The action performed from the Service server side is stored in the log database.

The proposed model constitutes scalable layers for the middleware architecture: at the lowest layer we have access to different computational entities (Smart phone, wearable computer, and tablet) and diverse sensors and actuators devices (RFID, camera, and marker). Next layer is an interface allowing two ways exchange between devices and middleware management layer. Third layer is autonomous middleware management layer which process the context modeling that includes the adaptive reasoning engine, a context knowledge base (KB), a context database and a context query engine. Next layer is a high level user context aware security interface between the middleware and the application layer. We propose to describe more precisely the Autonomous and adaptive middleware layer in which the ontology based reasoning engine is the brain of the context aware middleware. Ontology based Context Reasoning is a logical process which derives implicit context from the explicit context i.e. it derives deduce data from the external sensors and other information sources. The proposed reasoning engine uses context history for the adaptation process.

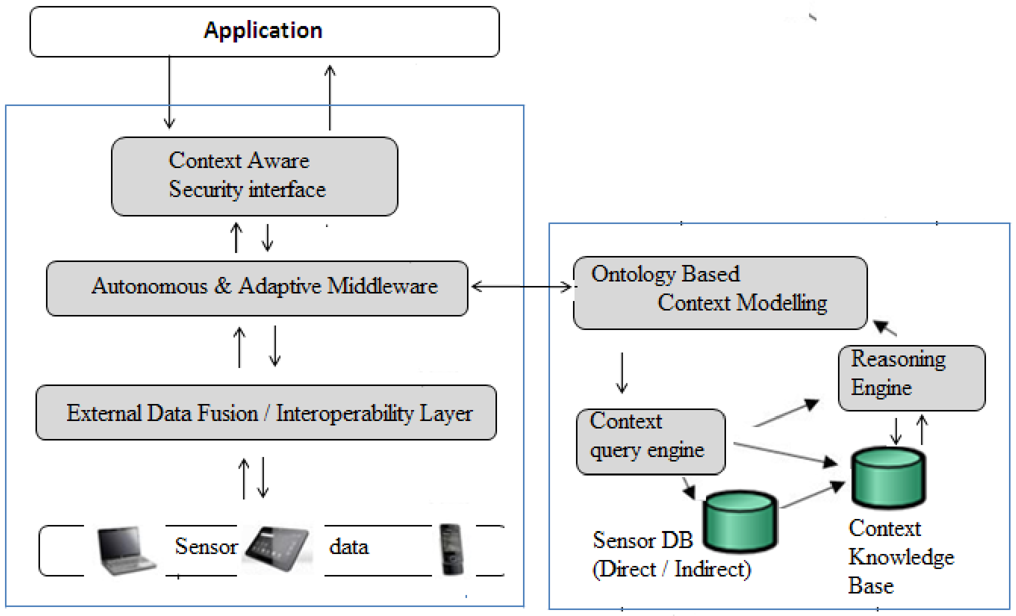
Application
Context Aware Security interface
Autonomous & Adaptive Middleware
External Data Fusion / Interoperability Layer
Sensor
data
Ontology Based Context Modelling
Reasoning Engine
Context query engine
Sensor DB (Direct / Indirect)
Context Knowledge Base

Fig.1. Ontology Based Context-Aware Middleware Architecture

The proposed middleware is a component oriented middleware which it automates context management by adding context aware components to the middleware and uses container characteristics to automate adaptation process as it is described in this section. Adaptation to context changes is achieved by following the main steps;

(1) Context data Collection using various sensors
(2) Context data Interpretation,
(3) Analyzing context data using reasoning engine
(4) Providing the Output to the users.

A. Sensor Layer

At the bottom are *sensors* that deliver raw sensor data. They may be wireless sensor networks, video cameras for tracking, RFID tags for identification, ultrasonic badges for location, or others. These produce raw sensor data, often pre-processed for saving communication cost as much as the (often power-constrained) sensor devices allow. These data are passed up one stage in the framework to the Sensor Data Fusion.

B. Sensors data fusion and Secure Information Interoperability Layer

It collects and integrates secure information from the divergent sensors and send appropriate information to the context Database;

C. Context database

It stores the context data (low level context) from sensors data fusion;

D. Context query engine

It handles the query from the application;

E. Context knowledge base

It stores the context model(environment context and users' context) by the ontology based model.

F. Reasoning Engine:

Context-aware systems must be able to perform context reasoning to facilitate dynamic adaptation to the changing environment, i.e. to be context-aware. Ontology based reasoning is proposed in the architecture. The proposed reasoning engine is organized as hybrid multi-level extendable engine: a strategy reasoning engine based on ontology reasoning and the detail reasoning engine based on decision tree reasoning. According to the collected raw data from sensors, reasoning engine exploring environment context and user context is able to take strategic appropriate decision. Then more precise reasoning can occur based on context decision tree. Here, a number of nodes can be combined to provide different decision paths based on information supplied by the sensors and context database. These nodes can communicate with context database, allowing up-to date information to be retrieved. The proposed model provides rule inference and acquisition capabilities based on *OWL* which can combine with semantic web rule-based systems to improve reasoning capabilities. Following are the functions performed by our proposed reasoning engine:

- Based on the reasoning rules that are defined by the user himself or the developer, context-aware systems can proactively provide context-aware services
- On the basis of context ontologies, reasoning engine deduces the high-level context from low-level context thereby verified through rule-based reasoning.

In this way we get a more bendable, robust, and efficient problem resolution in complex ambient intelligence situations.

G. User Security Layer

User security layer focuses on the access control issues. The goals of designing and developing access control mechanisms contingent upon various environmental and application-dependent contexts with secure provision for delegation of access control rights. This layer advocates a hybrid approach based on *Role, Status, Kerberos, Network RSKN authentication service by discretionary access control DAC, role-based Access Control RBAC and context-aware access control.*

This layer consists of *Security Manager* and *Context aware manager.*

1) ***Security Manager***

Security Manager runs single authentication and consists of sub-modules as follows:

• *Security Agent*: collects user authentication information when users log in the system or network.

• *Security Gateway*: Installed in security server, and processes user authentication with user authentication information of Security Agent.

2) ***Context-Aware Manager***

Context-Aware Manager performs the access control process and essentially composed of four processes: (i) *Identification* (ii) *Authentication* (iii) *Authorization* and (iv) *Access decision*. The four processes are performed in the any or all of the following sub-modules in Context aware manager,

- *Context-Aware Process:* related to access resource, access context and user permission,. It inquires authentication process with context-awareness.
- *Repository:* Saves context information collected during user authentication, intrusion detection and response time.

- *Role Based Access Control (RBAC)*: Accesses control settings for each resource.
- *Security Process Rule:* Defines user authentication procedures and roles.
- *User Authentication Analysis:* Analyzes the user's permission.
- *Access Context Analysis:* Analyzes user access network and time context.
- *Access Resource Analysis:* Analyzes access permission for the access resource.

i. *Data Flow between User and Security System*

By examining the data flow between the user and context-aware security system, the security manager transfers the information, which is acquired when the user logs in, such as *user ID, IP address, IP address of the resource being accessed, access time*, and other relevant information to a *context-aware manager*. The *context-aware manager* (security server) determines the security level based on the context information, then transfers security process (authentication procedure and access control information) and process Information (process blocking information) to the *security manager*. If the authentication succeeded, the user can proceed further.

ii. *Dynamic Analysis of Context Information*

A Context-aware security system consists of an *OSGi* service platform based bundles, thereby dynamically analysing context information for each different domain. By using the dynamic analysis, it allows the system to lessen the system overhead due to its flexible application of security service according to various field statuses.

H. Application Layer

This layer gathers information specific to the application allowing the designer to describe:

- All relevant contexts to which the application is sensitive by creating instances.
- Context-aware components belonging to the application by creating instances and binding them with the described relevant context using their properties.
- Interpretation rules for indirect context description, by creating instances of the class to describe interpretation methods and the property which enables to specify the indirect context to be deduced using context information.
- Reactive adaptation of the application, when a relevant context is detected, by describing the property which binds relevant context with the associated adaptation method class.
- Proactive adaptation of the application, by describing the property which binds relevant context with the policy to be invoked as instance, and the component ,which invokes the relevant service.

VI. Methodology

A. Design Considerations

In present computing environments, system should understand dynamic context information and take action consequently. Therefore to satisfy these necessities, it is necessary to separate context awareness and application development, that is to separate the underlying sensing, processing and high level context aware applications. The implementation of context-awareness is made possible by a combination of different technologies. First, the system needs appropriate data collection technologies for sensors and fusing technologies at a higher level for analysing and interpreting sensor data. Decoupling the sensing layer from applications is a key design principle for context-aware applications. To this end, software frameworks are needed to simplify and standardize the communication between applications, sensors and actuators. Moreover, to maximize the usefulness of context information for different services, designers need to create standard and widely accepted models with standard mechanisms in together for accessing these models for context information.
OWL language is chosen for representing information about objects and relationships relevant for the Context-aware service task. The main advantage of this context model is sharing the common structure of context and understanding the context information among users. This enables devices and services to enable semantic interoperability to meet the needs of the individual user. It also enables reuse of domain knowledge. Most importantly, it enables formal analysis of domain knowledge. The use of ontology will make our model independent of programming and application environment. In addition to the standardization of the structure of the context representation, it will help us to provide semantic descriptions and relationships of entities.

B. Context Modelling

1) Ontology-Based Context Modelling Approach

We have chosen ontology based models because they constitute a standard representation, reasoning and inferring scheme of knowledge. OWL language is chosen for representing information about objects and relationships relevant for the Context-aware service task. The main advantage of this context model is sharing common understanding of the structure of context information among users. This enables devices and services to enable semantic interoperability to meet the needs of the individual user. It also enables reuse of domain knowledge. Most importantly, it enables formal analysis of domain knowledge. The use of ontology will make our model independent of programming and application environment. In addition to the standardization of the structure of the context representation, it will help us to provide semantic descriptions and relationships of entities

2) Conceptualization

A class has a name and a set of properties that describe the characteristics of the class, Class is also called "concept", "type", "category" and "kind" in some ontology specification languages. The class can be instantiated by creating Individuals. Property describes an attribute of a class or an individual. A property can also be composed by other properties. For example "*GradeAssesment*" is based on "Evaluator", we define a property "*BasedOn*" for GradeAssessment and assign it to the class "Evaluator". Property is also referred to as "aspect", "attribute", "feature" or "characteristic". Relation: defines ways in which classes or individuals can be associated with each other. In our ontology definition model, the types of relations are predefined. To express specific relations, we need to use properties. Four types of relations are defined to connect classes and individuals:

a) *Subclass*: extends an abstract class to convey more concrete knowledge.
b) *PartOf*: means a class or an individual is a part of another class or individual.
c) *Complement*: expresses that the members of a class do not belong to another class; and the two classes together contain all the members in a given domain; and
d) *Equivalence*: means that two classes, individuals or properties are exactly the same.

Our context ontology modelling uses axiom such as *owl:subclass, owl:inverseOf, owl:unionOf, owl:disjointWith and owl:sameAs* which are provided in OWL as shown in the following example:

```
<owl: Class rdf:ID='Evaluator'>
<rdfs:subClassof>
<owl:Restriction>
<owl:onPropertyrdf:resource='Graduat'/>
<owl:toClassrdf:resource ='#Person'/>
<owl:classifiedAsrdfs:resource='ftp://305678/classification
#Reference'/>
</owl:Restriction>
</rdfs:subClassof>
</owl:Class>
```

We generated Ontology-based context metadata which is stored in the repository and is retrievable by the inference engine. The data which is generated by sensors provides the information used by service inference engine.

c. Building of ontology

The context ontology should be able to capture all of the characteristics of context information. Our context ontologies are divided into high-level and low-level ontologies as proposed by C.K.Law [17]. The high-level ontology is a generic ontology which captures basic context knowledge about the physical world environments. The low-level ontologies are a collection of domain-specific ontologies. The proposed model contains *Sensor Description Ontology*, *Direct Context Ontology*, *Indirect Context Ontology*, *User Context Ontology* and *Application level Ontology* and each ontology is associated to one another.

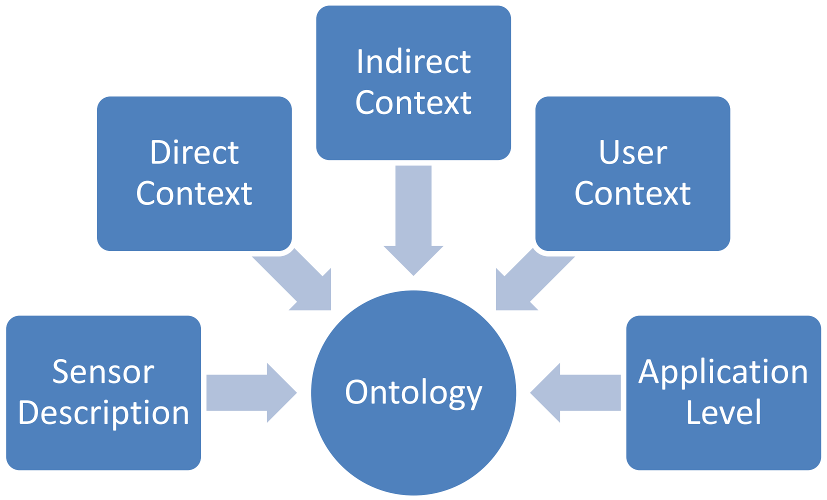

Fig. 2. The Proposed Ontology model

The *Sensor description ontology* is associated to the sensors description. This ontology contains information about sensors that the middleware can interact with. This information is created and updated easily by the middleware maintenance agent to enable the middleware to interact with new sensors. Sensors can be used by all applications running on top of this middleware. The direct context, Indirect context and user context ontology of Fig. 2 is associated to the context level. Context level supports interpretation rules and adaptation policies. Two types of adaptation are taken into account, reactive and proactive adaptations. Reactive adaptation consists in listening to changes in the execution context and reacting accordingly. Whereas proactive adaptation, which concerns service request, consists in anticipating the way in which the service is provided by changing the source code of the service policy by another policy (the same service can be provided using different policies) according to context changes. The *direct context* is captured directly from sensors, and the *indirect context* is interpreted from other contexts. The *user context* ontology specifies users of the system and their roles in an application. The *application level* ontology is associated to the application level, it gathers information specific to the application allowing the designer to describe. All relevant context to which the application is sensitive by creating instances of the *RelevantContext* class.

3) *Example Scenario*

From the research survey [19] it is found that context awareness can be tested in the domain of education,other areas in which it can be tested are as sport, health, business and administration.

To illustrate the use of the proposed model for the creation of context aware applications, we give in this section a simple example scenario for laboratory room management (figure 3) to provide appropriate exercises to the divergent participants of the lab. This room is equipped with indoor RFID sensors. Each student and faculty wears an RFID tag; this tag provides data to detect the role of participant. RFID readers are placed in the laboratory room to enable capturing signals emitted by each RFID tags. A master server connected to the network makes RFID information at the disposal of the middleware. When a person enters in the laboratory room, presence is detected, and then the related lab exercises, assignments are automatically transferred to the laptop or PDA. The laboratory room management application is made up of three components. The *Labsession* component is installed in the server, it contains all lab sessions in a semester. The *Labdetails* component is a plug-in that supports divergent devices of students and faculty members; it enables to download the laboratory exercise programs and all the related information for understanding the subject. The *ViewAgent* Component is a plug-in to divergent devices which enables to display a selected laboratory details in the user device. The laboratory room application is sensitive about user and user proximity in understanding the exercises. The latest teaching context is deduced from *StudentUnderstanding*. To enable the middleware to capture location information from RFID, the sensor instance (RFID) illustrated in Fig. 4 has to be described in the ontology.

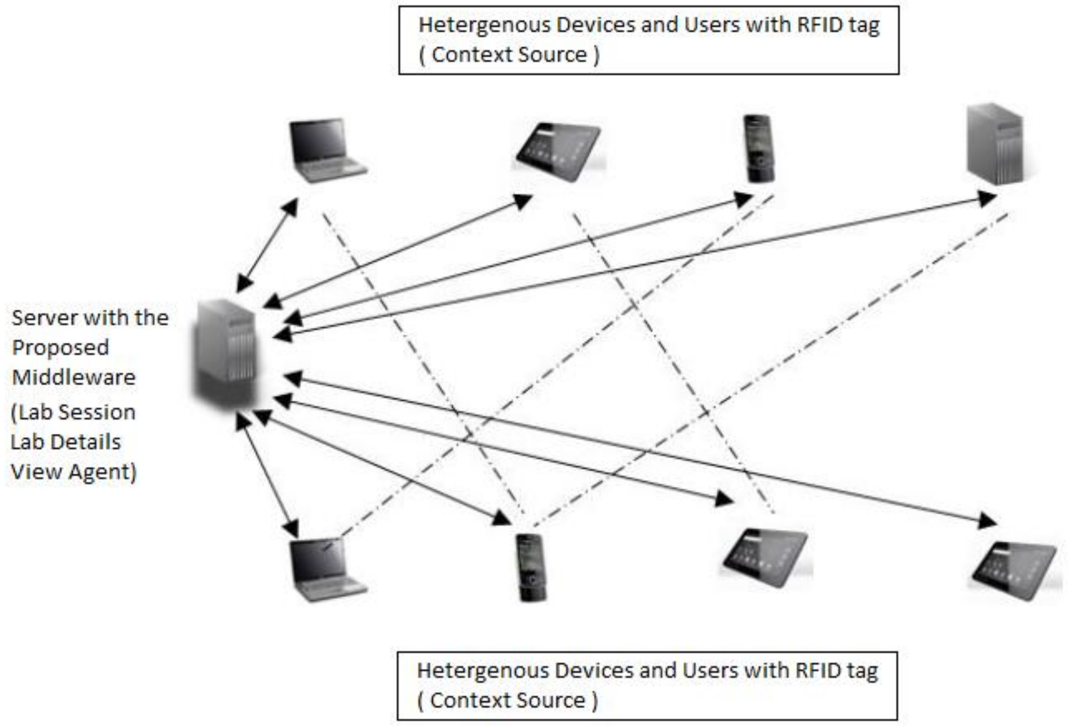
Hetergenous Devices and Users with RFID tag
(Context Source)
Server with the Proposed Middleware
(Lab Session
Lab Details
View Agent)
Hetergenous Devices and Users with RFID tag
(Context Source)

Fig. 3. Example Scenario Laboratory Room Management

The RFID sensor instance allows the middleware to interact with the master server, in order to detect the presence of a person in the laboratory room and give the details of person.

```
<Sensor rdf:ID="RFID">
<SensorName>RFIDSensor</SensorName>
<SensorSourcePath>http://www-inf.int-evry.fr/~belhanaf /Sensor
/RFID
</SensorSourcePath>
<SensorDataParam>SID</SensorDataParam>
<SensorParamType>Integer#Integer</SensorParamType> </Sensor>
```

Fig. 4. Description of the RFID sensor

The example scenario is sensitive about StudentUnderstanding which is captured directly from the RFID sensor and *ProximityToSubjectUnderstanding* which is interpreted from Stude*ntUnderstanding*. These two contexts are described in this ontology as illustrated in Fig 5. Interpretation policy belongs to the application level, its description is done in the third ontology.

```
<owl:Class rdf:ID="ProximityToSubjectUnderstanding">
 <rdfs:subClassOf rdf:resource="#Indirect"/></owl:Class>
 <owl:Class rdf:ID="StudentUnderstanding">
 <rdfs:subClassOf rdf:resource="#Direct"/>
<TakesDataFrom rdf:resource="#RFID"/>
</owl:Class>
```

Fig. 5. Example of Context Description

Fig. 6. describes interpretation rules which enable to interpret *ProximityToStudentUnderstanding*.

```
<HowDeduce rdf:ID="deduceProximity">
<HowDeduceDomaine>StudentUnderstanding
</HowDeduceDomaine>
<HowDeduceRange> ProximityToStudentUnderstanding
</HowDeduceRange>
<ExecuteDeduction rdf:resource="#nearUnderstanding"/>
</HowDeduce>
```

Fig. 6. Example of Indirect Context Interpretation

This context is deduced by comparing the student current understanding with the subject grade level using the *NearUnderstanding* method.

Fig. 7 illustrates description of the *NearUnderstanding* method as instance of the *Skill&Grade* class by specifying the path of the class containing the method source code, its name, the type of its parameters, and the type of the returned value. This method enables the middleware to determine whether a student understanding to the subject is clear in the laboratory with its *Skill* and *grade*.

```
< Skill&Grade rdf:ID="NearUnderstanding">
<SubjectAction>NearUnderstanding</SubjectAction>
<ParamType>varInteger#varInteger#Integer#Integer
</ParamType> <ReturnedType>String<ReturnedType>
 </Skill&Grade>
```

Fig. 7. Example of Interpretation method

Fig. 8 describes the relevant context and the adaptation policy which has to be invoked when this context is detected. This policy consists in opening the relevant student lab exercises when student log on into the system. This method has to be described as an instance of the *Skill&Grade* class

```
<RelevantContext rdf:ID="relevantContextToSubject">
<ContextName>ProximityToSubjectUnderstanding</ContextName>
<Operation>Equal</Operation> <Value>Near</Value>
</RelevantContext>
<HowAdapt rdf:ID="adaptProximityToSubjectUnderstanding">
<IfRelevantContext rdf:resource="#relevantContextToSubject"/>
<ExecuteAdaptation
rdf:resource="#openStudentExercise"/>
</HowAdapt>
```

Fig. 8. Example of Adaptation policy

Each component of the application is sensitive about a particular relevant context as illustrated in Fig.9 . The *LabDetails* component is sensitive about student subject and devices, whereas the *ViewAgent* component is sensitive about student proximity in subject understanding.

```
<Component rdf:ID="LabDetails">
<ComponentName>LabDetails</ComponentName>
 <AwareOf rdf:resource="#relevantInLaboratoryRoom"/>
</Component>
<Component rdf:ID="ViewAgent"> <ComponentName>
ViewAgent</ComponentName>
<AwareOf rdf:resource="#relevantContextToSubject"/>
</Component>
```

Fig.9 . Example of context-aware component description

4) *Evaluation of the model*

To evaluate the effectiveness and the performance on our proposed model, we conducted a questionnaire analysis as shown in the table 1. Students were asked to answer questions for four aspects: (1) effectiveness , (2) convenience, (3) usefulness, and (4) understanding using a 1-5 Likert scale with interval 1, which means 'very bad', 'bad', 'average', 'good', 'very good', respectively. Overall, the students replied highly positive to the questionnaire items as shown in Fig. 10. The reliability test (Cronbach's Alpha) shows high scores for effectiveness, convenience, usefulness and understanding.

Technical quality

TQ01	Interaction of the system
TQ02	Accessibility of the system
TQ03	context awareness
TQ04	Reliability of the system
TQ05	Performance of the system

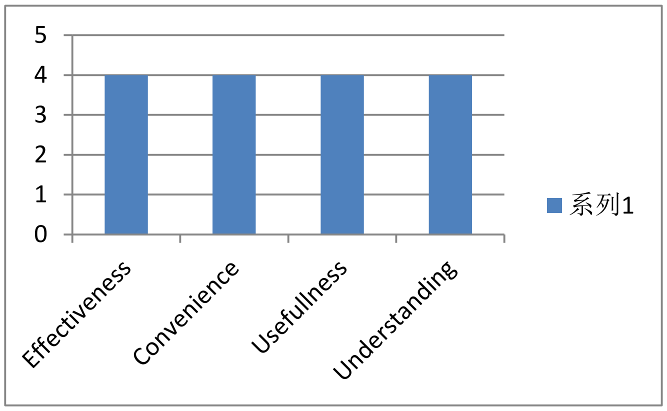

Fig.10 . Evaluation of the proposed model

VII. Conclusion

Context awareness helps applications running on mobile devices to react into changing environment conditions. An efficient context representation and a context aware middleware infrastructure are necessary for helping developers to create such type of applications.

In this paper, we proposed, a context aware middleware which provides ontology meta model for context description and application adaptation. The proposed model allows context and relevant context description. All the described data are interpreted in the middleware for relevant context detection. Relevant context is associated to an adaptation policy, the container has to apply this adaptation if a relevant context is detected. The proposed middleware takes into account two types of adaptations, reactive and proactive adaptations. These adaptations are carried out by the component container to which new controllers were added. Comparing to the existing context models solutions, the proposed ontology based middleware model adds the possibility to describe actions of adaptation and sensors from which data are collected. The relationships between different classes of the ontology are predefined by the proposed meta-model. These relationships allow the middleware to manage the collected context information and to adapt applications and services to context changes dynamically. Our future work aims to provide services to the users without ambiguous and uncertainty in the context awareness.

References

1. Hongyuan Wang, Rutvij Mehta, Lawrence Chung, Sam Supakkul, LiGuo Huang: “Rule-based context-aware adaptation: a goal-oriented approach,” Int. J. Pervasive Computing and Communications 8(3):pp. 279-299, 2012.
2. Bei Wang; Dongsheng Liu; Szemin Wong, A Context Information Ontology HierarchyModel for Tourism-oriented Mobile E-commerce, Journal of Software (1796217X),Vol. 7 Issue 8, p1751-1758. 8p Aug2012.
3. Lagares-Lemos, Miguel, Vasquez, Daniel Villanueva, Radzimski, Mateusz, Lemos, Angel Lagares and Berb ŝ, Juan Miguel Gómez, “RING: A Context Ontology for Communication Channel Rule-based Recommender System,”e meeting of the Proceedings of SeRSy, 2012.
4. Y. Jiang and H. Dong, “Uncertain context modeling of dimensional ontology using fuzzy subset theory,” in Proceedings of the 2nd International Conference on Scalable Uncertainty Management, pp. 256-269, 2008.
5. R.John and S.Coupland, “Type-2 Fuzzy Logic – A Historical View,” IEEE Computational Intelligence Magazine.vol. 2, pp.57-62, 2007.
6. S. J. H. Yang, J. Zhang, and I. Y. L. Chen, “A JESS-enabled context elicitation system for providing context-aware web services,” Expert Systems with Applications, Vol. 34, pp. 2254-2266,2008.
7. W. N. Schilit, ”A system architecture for context-aware mobile computing,” PhD Thesis, Columbia University, New York, 1995.
8. Strang, Th. and Linnhoff-Popien, C., 2004. “A Context Modeling Survey,” In 1st Int’l Workshop on Advanced Context Modelling, Reasoning and Management, pp. 34–41, Sep 2004.
9. T.R., Gruber, “A Translation Approach to Portable Ontology Specification,” Knowledge Acquisition, 5(2):199–220, Jun 1993.
10. H. Chen, F. Perich, T. Finin, and A. Joshi, “Soupa: Standard ontology for ubiquitous and pervasive applications,” in Mobile and Ubiquitous Systems: Networking and Services, 2004. MOBIQUITOUS 2004. The First Annual International Conference on. IEEE, 2004, pp. 258–267.
11. Tao Gu, Hung Keng Pung, Da Qing Zhang, “Toward an OSGi-Based Infrastructure for Context-Aware Applications,” IEEE Pervasive Computing, vol.03, no.4, pp.66-74, Oct-Dec, 2004.
12. M. Weiser, “The computer for the 21st century,” Scientific American, Vol. 265, pp. 94-104, 1991
13. Laura Maria Daniele,“Towards a Rule-based Approach for Context-Aware Applications,” Thesis for a Master of Science degree in Electronic Engineering from the University of Cagliari, May, 2006.
14. N.Georgalas; S.Ou; M Azmoodeh.; Kun Yang, "Towards a Model-Driven Approach for Ontology-Based Context-Aware Application Development: A Case Study," Model-Based Methodologies for Pervasive and Embedded Software, 2007. MOMPES '07. Fourth International Workshop on , vol. 21, no. 32,pp.31-31,March2007. doi: 10.1109/MOMPES.2007.18

15. Elenichristopoulou, Achilleskameas, C. Goumopoulos, "An Ontology-Based Context Management and Reasoning Process for Ubicomp Applications," soc-eusai '05 Proceedings of the 2005 Joint Conference on Smart Objects and Ambient Intelligence: Innovative Context-Aware Services: Usages and Technologies , pages 265 – 270, ACM New York, ny, USA ©2005 ,isbn:1-59593-304-2, 2005.
16. H.Chen, Finin T, Joshi A. An ontology for context-aware pervasive computing environments. Knowledge Engineering Review: Special Issue on Ontologies for Distributed Systems,vol.18(3): 197-207, 2004.
17. C. K. Law, "Using fuzzy numbers in education grading system," Fuzzy Sets and Systems, vol. 83, no. 3, pp. 311-323, 1996.
18. Mcheick, H.. Modeling Context Aware Features for Pervasive Computing. *Procedia Computer Science*, 37, pp.135-142, 2014.
19. Chambers, A., Models, AmI-Creator and A-Methodology for Ambient Intelligence Environments. *Journal of Software Engineering and Applications*, **7**, 311-346 2014.