Using Semantic Web for Ubiquitous Computing

Amine Tafat, Michele Courant, Beat Hirsbrunner, and Samir Sitaleb

Pervasive and Artificial Intelligence Group,
Computer Science Department,
Fribourg University
1700 Fribourg, Switzerland
http://diuf.unifr.ch/pai/

Abstract. Human interaction occurs always in a specific context and in a particular environment, and a common knowledge base about them is essential for understanding each other. By immersing computational system into physical world, ubiquitous computing bring us from traditional desktop computing interaction, to a new paradigm of interaction closer to humans one's in term of context and environment dependency, and knowledge sharing.
To tackle this problem, we present in this paper, XCM, a generic coordination model for Ubiquitous Computing. XCM is organized around a few abstract concepts (entity, environment, social law and port), and is expressed as an ontology by using semantic web languages. While the abstract concepts of XCM deal with environmental representation and context-dependency, the use of the semantic web language OWL allows us to achieve knowledge sharing and context reasoning within Ubiquitous Computing.

Keywords : Ubiquitous Computing, context-awareness, Ontology, Semantic Web, OWL.

1 Introduction

Computing is moving toward ubiquitous environments in which devices, software agents, and services are all expected to seamlessly integrate and cooperate in support for human objectives - anticipating needs, negotiating for service, acting on our behalf, and delivering services in anywhere any-time fashion [1].

Ubiquitous environments can be considered as physical environments saturated with computing and communication, yet gracefully integrated with human users. In the future our daily environment will contain a network of more or less specialized computational devices that interact among themselves and with us.[4]

However, the tendency of merging physical world and computational systems, requires an appropriate software infrastructure and development tools. In particular, from coordination and integration perspective, the use of computers in non-desktop environments leads to go beyond traditional desktop interaction paradigm.

When observing interaction between humans it appears that communication is influenced by context and environment. The situation in which the communication takes place provides a common ground. This common ground generates implicit conventions, which influence and to some extent set the rules for interaction and also provide a key to decode meaning of words and gestures [2] Moreover, a common knowledge base is essential for understanding each other.

Thus, in order to successfully embed ubiquitous computing capabilities in our everyday environments, we need to define a coordination paradigm that take into account the two humans interactions characteristic cited earlier: Context-awareness, and knowledge sharing.

In this paper we propose a generic coordination model XCM based on:

– Reflective approach, that models and represents physical world. Embedding physical world representation into applications allows to enhance ubiquitous computing components with contextual information, achieving context modelling and making applications context-aware.

– Semantic Web [12], to express the modeled contextual informations as an ontology through the Ontology Web Language[20]. The ontological representation of context allows us to achieve contextual knowledge sharing and context reasoning between applications partners.

The present paper is organized as follows. In the next section we overview the concept of coordination model, discuss requirements for Ubiquitous computing coordination model, and the gain obtained from using semantic web. Section3 describes the concepts of abstract model XCM, and its ontological representation with Ontology Web Language OWL. In Section 4 a conference scenario illustrating XCM model is presented. In section 5, we discuss related works, and in the last section we conclude this document.

2 Coordination and Coordination Model

Coordination can be defined as the process of managing dependencies between activities [5], or, in the field of Programming Languages, as the process of building programs by gluing together active pieces [6]. To formalize and better describe these interdependencies it is necessary to separate the two essential parts of a distributed application namely, computation and coordination. This sharp distinction is also the key idea of the paper of Gelernter and Carriero [6] where the authors propose a strict separation of these two concepts. The main idea is to identify the computation and coordination parts of a distributed application. Because these two parts usually interfere with each other, the semantics of distributed applications is difficult to understand.

A general coordination model is usually based on three components [7] :

– Coordination entities, as the processes or agents which are subject of coordination;
– Coordination medium, the actual space where coordination takes place;
– Coordination laws, to specify interdependencies between the active entities;

As Ubiquitous computing is nothing than computing in the context [2], it is essential to the intended coordination model to focus onto the context-sensitiveness of the manipulated entities. The model must represent the context, and enable sharing contextual knowledge between interacting entities.

In order to achieve this purpose, it requires contextual information to be represented in ways that are adequate for processing and reasoning. Semantic Web [12] provides a suitable technology to define ontologies for modeling context, providing common vocabularies for knowledge sharing. Our proposal is to use OWL language [20] to define a coordination model for Ubiquitous computing called XCM.

3 XCM, Generic Coordination Model

XCM is a coordination model designed to support the specificity of a Ubiquitous application. Its essential characteristics are:

– Genericity, which is obtained through a high level of abstraction based on the notion of entity and agent;
– Capacity to handle the dynamics of ubiquitous execution environments -either they are physical or virtual-, and the context-sensitivity of applications, thanks to the explicit notion of environment;
– Homogeneous management of the contextual dynamics of components by the unique formalism of social law attached to the notion of environment, and a mechanism of port allowing entities to interact both very flexibly and powerfully.

As a coordination model, XCM comes within P. Ciancarini's approach [7], and the vision of coordination proposed by T. Malone [5], while prolonging an experience of coordination platform development we had previously carried on [9]. Within this approach, however it adds on a theoretical component inspired by autopoiesis i.e. the modeling of living systems elaborated by F. Varela and H. Maturana [10]. The interest of autopoiesis heritage is double. First, it allows profiting from the specificity of the physical space for modeling mechanisms like the dynamic management - namely the construction and the maintenance- of organism frontiers. Second, it introduces a fundamental distinction between organisation (domain of control expression) and structure (domain of entity existence).

3.1 Entities

Everything is an entity in XCM. An entity e_i is defined by its structure, which is expressed as a recursive composition of entities $e_{i1}e_{in}$ -called components of e_i- and by its organisation.

When modeling the entity concept as an ontology, we define a class called `Entity`. This abstract class defines a set of properties that are common to all entities, which consists of `HasStructure` and `HasOrganisation`.
`Entity` classes have associated containment relationships. The relationships are defined by two related object properties called `HasComponents` and `IsComponentOf`. `HasComponent` property describes that the subject of this property is composed by the object of this property, and describes the subject of this property is a component of the object of this property. In the context of the OWL language, these two properties are defined as an inverse property of each other, as shown in the partial XCM ontology code bellow:

```
<owl:Class rdf:ID="Entity">
  <owl:UnionOf rdf:parseType="Collection">
   <owl:Class rdf:about="#AtomicEntity/>
   <owl:Class rdf:about="#CompoundEntity/>
  </owl:UnionOf>
 </owl:Class>

 <owl:ObjectProperty rdf:ID="HasComponents">
    <rdfs:domain rdf:resource="#Entity"/>
    <rdfs:range  rdf:resourcre="#Entity"/>
 </owl:ObjectProperty>

 <owl:ObjectProperty rdf:ID="IsComponentOf">
    <owl:inverseOf rdf:resource="#HasComponents"
    <rdfs:domain rdf:resource="#Entity"/>
    <rdfs:range  rdf:resourcre="#Entity"/>
 </owl:ObjectProperty>
```

Atomic entity. An entity, whose structure can not be decomposed, is called atomic; it denotes a pre-constructed element of the system. Whereas, the highest-level entity recursively containing all the other entities of the system, is called the universe of the system.

As atomic entity do not contain other entities, we introduce an abstract class called `AtomicEntity` which inherits all properties from its superclass `Entity` while adding restrictions on the range of `HasComponents` property. In `AtomicEntity` class, the cardinality of the property `HasComponents` is 0 indicating all instances of this class do not contain any other entity. On the other hand, `CompoundEntity` is introduced to represent a set of entities that contains at least one `Entity` class member. `CompoundEntity` inherits all properties from `Entity` class with restrictions on minimal cardinality of `HasComponents` property. Partial XCM ontology code is presented here:

```
 <owl:Class rdf:ID="AtomicEntity">
    <rdfs:subClassOf rdf:resource="#Entity"/>
    <rdfs:subClassOf>
     <owl:Restriction>
        <owl:OnProperty rdf:resource="#HasComponents"/>
        <owl:maxCardinality rdf:datatype="&xsd;nonNegativeInteger">0
        </owl:maxCardinality>
     </owl:Restriction>
    </rdfs:subClassof>
 </owl:Class>

 <owl:Class rdf:ID="CompoundEntity">
    <rdfs:subClassOf rdf:resource="#Entity"/>
    <rdfs:subClassOf>
     <owl:Restriction>
        <owl:OnProperty rdf:resource="#HasComponents"/>
        <owl:minCardinality rdf:datatype="&xsd;nonNegativeInteger">1
        </owl:minCardinality>
     </owl:Restriction>
    </rdfs:subClassof>
 </owl:Class>
<Entity rdf:ID="Universe" />
```

The organisation of an entity e_i specifies the rules governing the assembling of components in the structure and their dynamics. Then it characterizes the domain of the interactions that are applied to e_i. It is expressed as a set of rules called by extension the social laws of e_i

3.2 Environment & Social Laws

At a given moment of the existence of the system, every entity e_i -except the universe- therefore exists as a component of another entity e. The entity e is called *environment* of entity e_i.

We express this formally in XCM ontology by asserting that, if two instances of Entity Class, e_i and e are related by `IsComponentOf` object property instance, then e is an $environment$ of e_i. Therefore, we define a new class called `Environment` as equivalent to $CompoundEntity$, since every compound entity represent an environment for its components entities.

Thanks to its social laws, the environment e prescribes and determines the interactions between e_i and e, as well as between e_i and the e_j -i.e. they rule out the assembling and disassembling of e_i with the other components of e-. These laws also govern the input of e_i into e, and the output of e_i from e.

Let us for example consider the case of an antenna: its environment is its coverage area, and the entering (respectively leaving) of mobile devices into (respectively from) it is controlled by its social laws. When its social laws confer to an entity the capacity to initiate operations modifying its own structure (internal autonomy) or its relations with its environment (external autonomy), this entity is commonly called an agent.

These facts are expressed in our ontology by the following elements: We define $HasLaw$ the object property expressing the fact that environment e has social law L attached to its organisation; $LawOf$ the object property defined as inversed $HasLaw$ object property expressing that law L is related to environment e, which means that in order to interact together, the components of the environment e must exchange messages that are on conformity with the law L. Therefore, we define $L - Message$ as an object property, which relates the M messages that are on conformity with L law. We present here a partial XCM code on this concern:

```
<owl:Class rdf:ID="Environment">
    <owl:equivalentClass rdf:resource="#CompoundEntity"/>
</owl:class>

<owl:Class rdf:ID="Law">
    <rdfs:subClassOf rdf:resource="#Entity"/>
</owl:Class>

<owl:Class rdf:ID="Message">
    <rdfs:subClassOf rdf:resource="#Entity"/>
</owl:class>

<owl:ObjectProperty rdf:ID="HasLaw">
    <rdfs:domain rdf:resource="#Entity">
    <rdfs:range rdf:resource="#Law">
</owl:objectProperty>

<owl:ObjectProperty rdf:ID="LawOf">
    <rdfs:domain rdf:resource="#Law">
    <rdfs:range rdf:resource="#Entity">
</owl:objectProperty>

<owl:ObjectProperty rdf:ID="L-Message">
    <rdfs:domain rdf:resource="#Message">
    <rdfs:range rdf:resource="#Law">
</owl:objectProperty>
```

An entity can be tight to several environments. However, due to the enrooting of ”ubiquitous” entities in the physical space, an entity can not be physically present in two different environment in the same time. This means it can be active at the most in one environment and ”virtually” or ”sensorially” present in other environments (cf.Ports). To express this property, we introduce in our ontology the two object properties $IsVirtuallyPresentIn$ and $IsPhysicallyPresentIn$, to express if entity is physically or virtually present in environment.

The notion of environment then encompasses within a single concept all the semantic diversity of the ubiquitous application components: a social semantics, inherited from coordination in general, and a physical semantics of entities, which becomes essential as soon as the entities are evolving onto -for example mobile- devices subjected to the laws of the physical space.

By social semantics, we mean the capacity of an entity to belong to a social structure, such as a group of entities taht it is interacting with (for instance a person belongs to a group of persons with which it is presently in meeting).

The XCM model supports multiple organisational linking of an entity (for instance a person is linked to a football club as member, but also and mainly to a company in the name of which it is predominantly acting during the meeting).

By physical semantics,we namely mean the impossibility for an agent to act in two environments at the same time, or to be ”teleported” from one environment to another . An entity can however remain ” aware of ” another environment than the one in which it is active. As we will see further, it can open some specific communication channels in this environment, thus implementing a remote perception mechanism.

Top

3.3 Ports

A port is a special type of entity dedicated to communication between entities. A port p has the specificity to be generally active while being coupled to an agent a_i, which is the port's master.

We define therefore in our ontology an Agent as subclass of $Entity$ class, with additional restrictions on the range of `HasPort` object property. In `Agent` class, the cardinality of the property `HasPorts` must be at least equal to 1 indicating all instances of this class has ability to communicate with external world with at least one port.

The coupling between a_i and p is obtained through a special type of composition called interface, which is specified by social laws (of p and a, and of their common environment). This is expressed in our ontology by adding `InterfaceComposition` class as subclass of Law class, and by coupling between an individual from $Entity$ class and individual from $Port$ class, through `HasInterfaceComposition` objec property.

These compositional laws define how the port is assembled to its master, for example maintained versus not maintained by master's movement, linked by ownership, or by usage, etc. They may also define the modalities of using the port (in terms of communication protocol, of bandwidth, etc). For answering ubiquitous computing needs, we also distinguish removable and irremovable ports.

Example: For a human agent, a mobile phone is a removable port, whereas an audio-prosthesis is irremovable. A pair of glasses is somewhere in between, obeying to coupling laws, that are stronger than the phone's ones, but looser than the prosthesis ones. An agent a may be coupled to several ports. It can acquire ports, and dissociate itself from ports dynamically. The agent-port assembling and disassembling procedures are triggered either explicitly, by an initiative of A, or implicitly by the entrance into a new environment, or by the environment dynamics, which may for example welcome new ports, which are automatically coupled to A.
Partial XCM ontology definition about ports is presented here:

```
<owl:Class rdf:ID="Port">
   <rdfs:subClassOf rdf:resource="#Entity"/>
   <rdfs:subClassOf>
    <owl:restriction>
     <owl:OnProperty rdf:resource="#HasOwner">
     <owl:minCardinality rdf:datatype="&xsd;nonNegativeInteger">1
     </owl:minCardinality>
    </owl:restriction>
   </rdfs:subClassOf>
</owl:Class>

<owl:Class rdf:ID="Agent">
   <rdfs:subClassOf rdf:resource="#Entity"/>
   <rdfs:subClassOf>
    <owl:Restriction>
     <owl:OnProperty rdf:resource="#HasPort"/>
     <owl:minCardinality rdf:datatype="&xsd;nonNegativeInteger">1
     </owl:minCardinality>
    </owl:Restriction>
   </rdfs:subClassof>
</owl:Class>

<owl:ObjectProperty rdf:ID="HasPort">
   <rdfs:domain rdf:resource="#Entity"/>
```

```
    <rdfs:range  rdf:resourcre="#Entity"/>
 </owl:ObjectProperty>

 <owl:ObjectProperty rdf:ID="HasOwner">
    <rdfs:domain rdf:resource="#Port"/>
    <rdfs:range rdf:resource="#Entity/>
 </owl:ObjectProperty>
```

The notion of port is then a fundamental mechanism, which confers to XCM the ability to coordinate context-sensitive entities. This context-awareness is the central characteristics of application components in ubiquitous computing.

4 Case study: Conference Scenario

to illustrate the XCM model described above, a conference scenario is presented. Three colleagues from the same research group are registered in a conference, with multiple sessions (four sessions for example, with each session holding a set of presentations) held in alternative rooms, are trying to attend different presentations.
Each individual obtains a hand held device(PDA) while he gives his personal details, as well as his research interests, at the registration desk. The PDA automatically displays a copy of the conference schedule highlighting the paper tracks which are of interest to each person. in our case, as the three colleagues belong to the same research group, they would have the same highlighted presentation sessions, which they would like to attend. To get a maximum outcome from the conference, each of the three individuals should attend a different presentation.
When each individual holding a PDA walks in a conference room, he is automatically picked up by the network and his name is added straight away to the attendents list.

The whole conference space including the four conference rooms represents the physical space. When expressing this scenario within XCM model, it is modelized as environmental compound entities ($ConferenceSpace$, $ConferenceRoom1 \ldots ConferenceRoom4$, each including its components.
The research group, to which belong the three colleagues, is represented as virtual (or represents a virtual...) environmental entity $ResearchGroup$.
The three colleagues are represented as three Agents $ResearchGroupMember1$, $ResearchGroupMember2$, $ResearchGroupMember3$. Each agent is represented within two environments: a physical environment which is the conference room in which the person is attending a presentation, and a virtual environment which is the research group to which the three colleagues belong.

```
<XCM:Environment rdf:ID="ConferenceSpace" >
   <XCM:HasComponent rdf:ID="ConferenceRoom1"/>
   ...
   <XCM:HasComponent rdf:ID="ConferenceRoom4"/>
</XCM:Environment>

<XCM:Environment rdf:ID="ConferenceRoom1" >
   <XCM:IsComponentOf rdf:ID="ConferenceSpace">
   <XCM:HasPort rdf:ID="SpotPointCR1">
</XCM:Environment>

...
<XCM:Environment rdf:ID="ConferenceRoom4" >
   <XCM:IsComponentOf rdf:ID="ConferenceSpace">
   <XCM:HasPort rdf:ID="SpotPointCR4">
```

```
</XCM:Environment>

<XCM:Environment rdf:ID="ResearchGroup" >
   <XCM:HasComponent rdf:ID="ResearchGroupMember1"/>
   ...
   <XCM:HasComponent rdf:ID="ResearchGroupMember3"/>
</XCM:Environment>

<XCM:Agent rdf:ID="ResearchGroupMember1">
<XCM:HasPort rdf:ID="PDA1" />
<XCM:IsComponentOf rdf:ID="ResearchGroup" />
</XCM:Agent>
...
<xcm:Agent rdf:ID="ResearchGroupMember3">
<XCM:HasPort rdf:ID="PDA3" />
<XCM:IsComponentOf rdf:ID="ResearchGroup" />
</XCM:Agent>
```

The social law governing the virtual environnement (the research team) is the fact of attending different presentations in order to maximize the outcome of the research group from the conference, which represents the interaction rule between individuals within this virtual environment. If an individual (eg.$ResearchGroupMember1$ chooses to attend a presentation taking place in $ConferenceRoom1$for example, the other individuals will have to choose another highlighted presentation related to their research intereset held in another conference room. This interaction rule is formalised in XCM as follow:

```
<XCM:Law rdf:ID="ExclusivPresenceInConf">
<XCM:LawOf resource:about="#ResearchGroup">
<owl:restriction>
 <owl:OnProperty rdf:resource="#IsPresentIn>
 <rdfs:domaine rdf:resource="#ResearchGroup"/>
 <rdfs:range rdf:resource="#ConferenceSpace"/>
 <rdf:type rdf:resource="&owl;InverseFunctionalProperty" />
 </owl:OnProperty>
</owl:restriction>

  ...
</XCM:Law>


<XCM:Environment rdf:about="#ResearchGroup" >
<XCM:HasLaw rdf:about="#ExclusivePresenceInConf"'>

<owl:ObjectProperty rdf:ID="IsPresentIn">
  <owl:EquivalentProperty rdf:resource="#HasComponent"/>
</owl:ObjectProperty>

</XCM:Environment>
```

The $ExclusivPresenceInConf$ law restricts the $IsPresentIn$ property (which is declared as equivalent to $HasComponent$ XCM property) with the $InverseFunctionalProperty$ property. This means that two members of $ResearchGroup$ cannot be located in the same conference room.

Top

Thus, when $ResearchGroupMember1$ joins $RoomConference1$ in order to attend a presentation session, the above fact is generated and is added to the contextual information about environment:

```
<owl:Thing rdf:about="#ResearchGroupMember1">
  <IsPresentIn rdf:resource="#ConferenceRoom1" />
</owl:Thing>
```

and this session is no more highligted in $ResearchGroupMember2$'s and $ResearchGroupMember3$'s PDAs, therefore, the they can only choose from the remainning highlighted presentations held in $ConferenceRoom2$ or $ConferenceRoom3$ according to the $ExclusivPresenceInConf$ law related to $ResearchGroup$ environment.

This example shows how XCM integrates the coordination aspect within pervasive computing. The definition of the environment within XCM model as a basic concept has allowed the information collected from the environment to be integrated in the application and to be exploited in the coordination process. The interaction between agents occurs in an implicit way based on both the social law expressed within their environment and the information gathered from it.

5 Related Works

Number of interesting frameworks are investigating Ubiquitous computing research, such as Context-Toolkit [11], Cooltown [17], Intelligent Room [16], OneWorld [15], EventHeap [8]. These systems use ad hoc representations of context knowledge, while GAIA [13], Cobra [14] and Semantic Gadgets [3], explore the use of ontology to represent context knowledge. EventHeap[8] is the first tuplespace coordination model intended to ubiquitous computing rooms.
Our contribution regarding theses systems is a generic approach for coordination in ubiquitous computing based on a global representation of context and physical world, and the use of Semantic Web languages to formalize ontologies of context, providing an explicit representation of contexts for reasoning and knowledge sharing. The proposed XCM model is used in UbiDev[19] framework and CB-Sec[18] framework.

6 Conclusion

Ontologies and explicit context representation are key requirements for realizing ubiquitous systems. Our works show that the newly emerged Web Ontology Language OWL is suitable for building a common knowledge representation for Ubiquitous computing to deal with context-aware interactions.

Through some abstract concepts -entity, environment, social laws and port- XCM takes place in a layer architecture allowing to apprehend in a conceptually simple and homogeneous way the diversity and the dynamics of Ubiquitous application universes. It integrates in particular the immersion of the application components within the physical universe, and the context-sensitiveness required by the ubiquitous applications. XCM model makes computer's interaction closer to humans one's in term of context and environment dependency, and knowledge sharing.

References

1. M.Weiser The computer for the 21st century. Scientific American, 265(30):94,104, 1991.
2. Albercht Schmidt, Ubiquitous Computing- Computing in Context, PHD thesis, Computing Department, Lancaster University, UK, 2002.
3. Ora Lassila, Mark Adler: Semantic Gadgets: Ubiquitous Computing Meets the Semantic Web, in: Dieter Fensel et al (eds.): Spinning the Semantic Web, pp.363-376, MIT Press, 2003
4. Lyytinen, Kalle, and Yoo Youngjin. Issues and Challenges in Ubiquitous Computing. Communications of the ACM 45(12): 62-65, 2002.
5. T.W. Malone and K. Crowston. The Interdisciplinary Study of Coordination. ACM Computing Surveys, 26(1):87-119, March 1994.

6. N. Carriero and D. Gelernter. Coordination Languages and Their Significance. Communications of the ACM, 35(2):97-107, February 1992.
7. P. Ciancarini, F. Arbab and C. Hankin: Coordination languages for parallel programming. Parallel Computing, 24 (7):989-1004, 1998.
8. Bradley Earl Johanson, *Application Coordination Infrastructure for Ubiquitous Computing Rooms*, PHD thesis, Stanford University 2003.
9. M. Schumacher: Objective Coordination in Multi Agent Systems Engineering. Springer Verlag. LNAI 2039, 2001. (also published as PhD Thesis, Dept of Informatics, University of Fribourg, Suisse).
10. F. Varela and H. Maturana. Autopoiesis and Cognition: The realization of the Living. Boston Studies in the Philosophy of Science in Cohen, Robert S., and Marx W. Wartofsky (eds.) , Vol. 42, Dordecht (Holland): D. Reidel Publishing Co., 1980.
11. Daniel Salber, Anind K. Dey, and Gregory D. Abowd. The context toolkit: Aiding the development of context-enabled applications. In CHI, pages 434441, 1999.
12. Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. Scientific American, May 2001.
13. Anand Ranganathan, Robert E. McGrath, Roy Campbell, and Dennis M. Mickunas. Ontologies in a pervasive com-puting environment. Workshop on Ontologies in Distributed Systems, IJCAI 2003, 2003.
14. Harry Chen et al., *An Ontology for Context-Aware Pervasive Computing Environments*, Workshop on Ontologies and Distributed Systems, August 2003.
15. Grimm, R., et al. A System Architecture for Pervasive Computing. in 9th ACM SIGOPS European Workshop. 2000. Kolding, Denmark.
16. Coen, M.H. A prototype intelligent environment. in Cooperative Buildings Integrating Information, Organization, and Architecture First International Workshop CoBuild'98 Proceedings. 1998. Darmstadt, Germany: Berlin, Germany : Springer-Verlag, 1998.
17. Kindberg, T., et al. People, places, things: Web presence for the real world. in Third IEEE Workshop on Mobile Computing Systems and Applications. 2000. Los Alamitos CA USA: Los Alamitos, CA, USA : IEEE Comput. Soc, 2000.
18. S. K. Mostfaoui, A. Tafat-Bouzid and B. Hirsbrunner. Using Context Information for Service Discovery and Composition. Proceedings of the Fifth International Conference on Information Integration and Web-based Applications and Services, iiWAS'03, Jakarta, Indonesia, 15 - 17 September 2003. pp. 129-138.
19. S. Maffioletti, S.Kouadri M. and B. Hirsbrunner . Automatic Resource and Service Management for Ubiquitous Computing Environments. to Appear in Middleware Support for Pervaisve Computing Workshop (at PerCom '04), PerWare '04, Orlando, Florida USA, 14 March 2004.
20. Sean Bechhofer, Frank van Harmelen, Jim Hendler, Ian Horrocks, Deborah L. McGuinness, Peter F. Patel-Schneider, and Lynn Andrea Stein. OWL Web Ontology Language Reference, w3c recommendation, 10 February 2004.