Ontology Based Context Modeling and Reasoning using OWL

Xiao Hang Wang[1, 2], Da Qing Zhang[1], Tao Gu[1, 2], Hung Keng Pung [2]
[1] *Institute for Infocomm Research, Singapore 119613*
[2] *School of Computing, National University of Singapore, Singapore 119260*
xwang@i2r.a-star.edu.sg, gutao@comp.nus.edu.sg,daqing@i2r.a-star.edu.sg, punghk@comp.nus.edu.sg

Abstract

In this paper we propose an OWL encoded context ontology (CONON) for modeling context in pervasive computing environments, and for supporting logic-based context reasoning. CONON provides an upper context ontology that captures general concepts about basic context, and also provides extensibility for adding domain-specific ontology in a hierarchical manner. Based on this context ontology, we have studied the use of logic reasoning to check the consistency of context information, and to reason over low-level, explicit context to derive high-level, implicit context. By giving a performance study for our prototype, we quantitatively evaluate the feasibility of logic based context reasoning for non-time-critical applications in pervasive computing environments, where we always have to deal carefully with the limitation of computational resources.

1. Introduction

Recent years have witnessed rapid advances in the enabling technologies for pervasive computing. It is widely acknowledged that an important step in pervasive computing is context-awareness. Computational entities in pervasive environments need to be context-aware so that they can adapt themselves to changing situations. With the advance of context aware computing, there is a increasing need for developing formal context models to facilitate context representation, context sharing and semantic interoperability of heterogeneous systems.

In previous works, both informal and formal context models have been proposed. Informal context models are often based on proprietary representation schemes which have no facilities to ease shared understanding about context between different systems. Among systems with informal context models, Context Toolkit [1] represents context in form of attribute-value tuples, and Cooltown [2] proposed a Web based model of context in which each object has a corresponding Web description. Formal context models commonly employ formal modeling approaches to manipulate context. Karen et al. [3] model context using both ER and UML models; context can be easily managed with relational databases. Anand et al.[4] represented context in Gaia system as first-order predicates written in DAML+OIL. Existing formal context models support formality and address a certain level of context reasoning. However, none of them has addressed formal knowledge sharing, or has shown a quantitative evaluation for the feasibility of context reasoning in pervasive computing environments, where we always have to face resource-constraint devices.

In this paper, we present an ontology-based formal context model to address critical issues including formal context representation, knowledge sharing and logic based context reasoning. We will present the detailed design of our context model and logic based context reasoning scheme. Through performance analysis, we will show a quantitative evaluation for context reasoning in pervasive computing environments. The rest of this paper is divided into five sections. In section 2, we introduce ontology definitions and Semantic Web. In section 3 we describe the design of our context model (CONON). Section 4 shows how logic based context reasoning can be used to enhance context-awareness. Section 5 describes our prototype implementation; followed by the performance evaluation. Section 6 summarizes this paper.

2. Ontology and Semantic Web

The term "ontology" has a long history in philosophy, in which it refers to the subject of existence. In the context of knowledge management, ontology is referred as the shared understanding of some domains, which is often conceived as a set of entities, relations, functions, axioms and instances.

There are several reasons for developing context models based on ontology:

- **Knowledge Sharing**. The use of context ontology enables computational entities such as agents and services in pervasive computing environments to have a common set of concepts about context while interacting with one another.
- **Logic Inference.** Based on ontology, context-aware computing can exploit various existing logic

reasoning mechanisms to deduce high-level, conceptual context from low-level, raw context, and to check and solve inconsistent context knowledge due to imperfect sensing.

- **Knowledge Reuse.** By reusing well-defined Web ontologies of different domains (e.g., temporal and spatial ontology), we can compose large-scale context ontology without starting from scratch.

Semantic Web [5] is an effort that has been going on in the W3C to provide richer and explicit descriptions of Web resources. The essence of SW is a set of standards for exchanging machine-understandable information. Among these standards, Resource Description Framework (RDF) provides data model specifications and XML-based serialization syntax, Web Ontology Language (OWL) [6] enables the definition of domain ontologies and sharing of domain vocabularies. OWL is modeled through an object-oriented approach, and the structure of a domain is described in terms of classes and properties. From a formal point of view, OWL can be seen to be equivalent to description logic (DL), which allows OWL to exploit the considerable existing body of DL reasoning including class consistency and consumption, and other ontological reasoning.

We believe that Web ontology and other Semantic Web technologies can also be employed in modeling and reasoning about context information in pervasive computing environments.

3. CONON: The Context Ontology

In this section we present an extensible CONtext ONtology (CONON) for modeling context in pervasive computing environments.

Due to evolving nature of context aware computing, completely formalizing all context information is likely to be an in-surmountable task. However, we found that location, user, activity and computational entity are most fundamental context for capturing the information about the executing situation. These contextual entities not only form the skeleton of context, but also act as indices into associated information. The objectives of our context model include modeling a set of upper-level entities, and providing flexible extensibility to add specific concepts in different application domains.

In realistic pervasive computing environments, applications and services are usually grouped as a collection of sub-domains for different intelligent environments (e.g., home, office or vehicle). Context in each domain shares common concepts that can be modeled using a general context model, while differs significantly in detailed features. Therefore, the separation of application domains encourages the reuse of general concepts, and provides a flexible interface for defining application-specific knowledge. We divide our context model into upper ontology and specific ontology. The upper ontology is a high-level ontology which captures general features of basic contextual entities. Specific ontology is a collection of ontology set which define the details of general concepts and their features in each sub-domain.

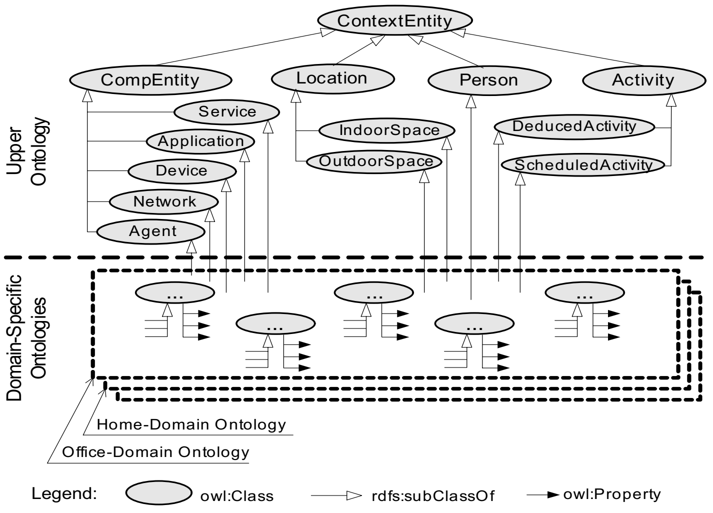

Figure 1. Partial Definition of CONON upper ontology

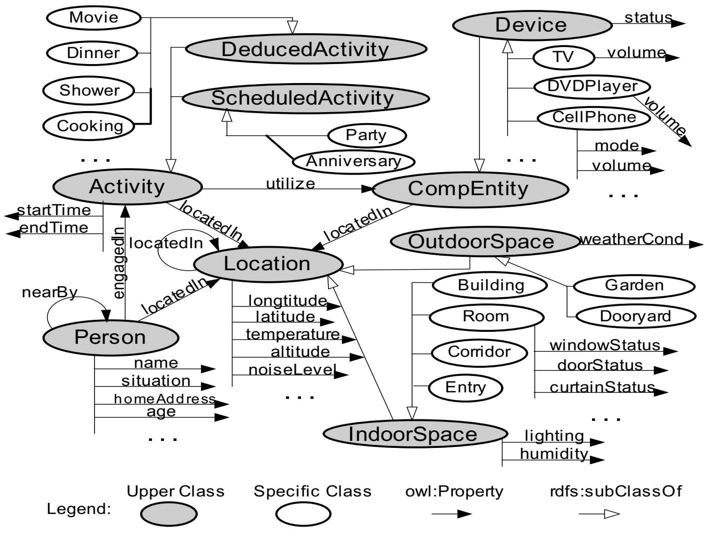

Figure 2. Partial definition of a specific ontology for home domain

Figure 1 shows the upper context ontology (the partial OWL serialization is show in figure 3). The context model is structured around a set of abstract entities, each describing a physical or conceptual object including *Person*, *Activity*, Computational Entity (*CompEntity*) and *Location*, as well as a set of abstract sub-classes. Each entity is associated with its attributes (represented in *owl:DatatypeProperty*) and relations with other entities (represented in *owl:ObjectProperty*). The built-in OWL property *owl:subClassOf* allows for hierarchically structuring sub-class entities, thus providing extensions to add new concepts that are required in a specific domain.

Figure 2 shows a partial definition of specific ontology for a smart home application domain. Besides general classes defined in CONON upper ontology, a number of concrete sub-classes are defined to model specific context in a given environment (e.g., the abstract class *IndoorSpace* of home domain is classified into four sub-classes *Building*, *Room*, *Corridor* and *Entry*).

```
<owl:Class rdf:ID="ContextEntity"/>
 <owl:Class rdf:ID="Location">
   <rdfs:subClassOf rdf:resource="#ContextEntity"/>
 </owl:Class>
 <owl:ObjectProperty rdf:ID="longtitude">
   <rdf:type rdf:resource="FunctionalProperty">
   <rdfs:domain rdf:resource="Location">
   <rdfs:range rdf:resource="xsd:double">
 </owl:ObjectProperty> ...
<owl:Class rdf:ID="IndoorSpace">
   <rdfs:subClassOf rdf:resource="#Location"/>
   <owl:disjointWith rdf:resource="#OutdoorSpace"/>
 </owl:Class>
 <owl:ObjectProperty rdf:ID="locatedIn">
   <rdf:type="owl:TransitiveProperty"/>
   <rdfs:domain rdf:resource="#Entity"/>
   <rdfs:range rdf:resource="#Location"/>
   <owl:inverseOf rdf:resource="#contains "/>
  </owl:ObjectProperty>  ...
```
Figure 3. Partial OWL serialization of the upper ontology

4. Context Reasoning

When taking a formal approach to model context, context can be processed with logical reasoning mechanisms. The use of context reasoning has two folds: Checking the consistency of context, and deducing high-level, implicit context from low-level, explicit context.

To explain the role of context reasoning in context-aware computing, we present a smart phone scenario in which a mobile phone can adapt to a user's current situation. By defining preference profiles, users can customize the behaviors of the augmented mobile phone. For example, when the user is sleeping in the bedroom or taking a shower in the bathroom, incoming calls are forwarded to voice mail box; when the user is cooking in the kitchen or watching TV in the living room, the volume of the ring is turned up; when the user is having dinner with the family in the dining room, the phone is set to vibrate mode. Obviously, high-level context can not be directly acquired from sensors; it is reasoned from sensor-driven, low-level context such as physical location and environmental information.

In this section, we will describe context reasoning based on CONON to demonstrate the key feature of the ontology based context model. We choose to implement context reasoning by using first-order predicates. The structure of the first-order predicate has tree fields - a subject an object and a verb. For example, the physical location context "Wang is located in the bed room" can be described as *(Wang, locatedIn, Bedroom)*. We believe that logics are very powerful tools for reasoning with context knowledge, and they are sufficient for general pervasive context-aware systems as is demonstrated later.

The reasoning tasks in our work are grouped into two categories: ontology reasoning using description logic, and user-defined reasoning using first-order logic.

4.1. Ontology Reasoning

Description Logic allows specifying a terminological hierarchy using a restricted set of first-order formulas. The equivalence of OWL and description logic allows OWL to exploit the considerable existing body of DL reasoning fulfill important logical requirements. These requirements include concept satisfiability, class subsumption, class consistency, and instance checking.

Table 1 shows a sub-set of reasoning rules that support OWL-Lite entailed semantics.

Table 1. Parts of OWL ontology reasoning rules

Transitive-Property	(?P rdf:type owl:TransitiveProperty) $\wedge$ (?A ?P ?B) $\wedge$ (?B ?P ?C) $\Rightarrow$ (?A ?P ?C)
subClassOf	(?a rdfs:subClassOf ?b) $\wedge$ (?b rdfs:subClassOf ?c) $\Rightarrow$ (?a rdfs:subClassOf ?c)
subProperty-Of	(?a rdfs:subPropertyOf ?b) $\wedge$ (?b rdfs:subPropertyOf ?c) $\Rightarrow$ (?a rdfs:subPropertyOf ?c)
disjointWith	(?C owl:disjointWith ?D) $\wedge$ (?X rdf:type ?C) $\wedge$ (?Y rdf:type ?D) $\Rightarrow$ (?X owl:differentFrom ?Y)
inverseOf	(?P owl:inverseOf ?Q) $\wedge$ (?X ?P ?Y) $\Rightarrow$ (?Y ?Q ?X)

In addition, ontology reasoning is also useful in other aspects of context aware computing. For example, in the example context ontology described in previous section, we define the relation '*locatedIn*' between a '*ContextEntity*' and a '*Location*' as an '*owl:TransitiveProperty*' relation, and the relation '*contains*' as the '*inverse property*' of '*locatedIn*'. Therefore, we can make use of the rules entailed by OWL to reason with physical location. An example result is shown in Table 3. Explicit context is acquired from context sources directly, while implicit context is the additional information deduced from explicit context. For example, knowing the user '*Wang*' is currently '*locatedIn*' the room '*Bedroom*', which in turn is a part of the '*Home*' building, description logic can be used to conclude that '*Wang*' is located in '*Home*' building as the spatial relation '*locatedIn*' is transitive.

Table 2. Reasoning about location using ontology

INPUT	DL Reasoning Rules	(?P rdf:type owl:TransitiveProperty) ∧ (?A ?P ?B) ∧ (?B ?P ?C) ⇒ (?A ?P ?C)
		(?P owl:inverseOf ?Q) ∧ (?X ?P ?Y) ⇒ (?Y ?Q ?X)
	Explicit Context	<owl:ObjectProperty rdf:ID="locatedIn"> <rdf:type="owlTransitiveProperty"/> <owl:inverseOf rdf:resource="#contains"/> </owl:ObjectProperty> <Person rdf:ID="Wang"> <locatedIn rdf:resource="#Bedroom"/> </Person > < Room rdf:ID="Bedroom"> < locatedIn rdf:resource="#Home"/> </ Room>
OUTPUT	Implicit Context	<Person rdf:ID="Wang"> <locatedIn rdf:resource="#Home"/> </Person > <Building rdf:ID="Home"> < contains rdf:resource="#Bedroom"/> < contains rdf:resource="#Wang"/> </Building> <Room rdf:ID="Bedroom"> < contains rdf:resource="#Wang"/> </Room>

Table 3. User-defined context reasoning rules

Situation	Reasoning Rules
Sleeping	(?u locatedIn Bedroom) ∧ (Bedroom lightLevel LOW) ∧ (Bedroom drapeStatus CLOSED) ⇒ (?u situation SLEEPING)
Shower-ing	(?u locatedIn Bathroom) ∧ (WaterHeater locatedIn Bathroom) ∧ (Bathroom doorStatus CLOSED) ∧ (WaterHeater status ON) ⇒ (?u situation SHOWERING)
Cooking	(?u locatedIn Kitchen) ∧ (ElectricOven locatedIn Kitchen) ∧ (ElectricOven status ON) ⇒ (?u situation COOKING)
Watching-TV	(?u locatedIn LivingRoom) ∧ (TVSet locatedIn LivingRoom) ∧ (TVSet status ON) ⇒ (?u situation WATCHINGTV)
Having-Dinner	(?u locatedIn DiningRoom) ∧ (?v locatedIn DiningRoom) ∧ (?u owl:differentFrom ?v) ⇒ (?u situation HAVINGDINNER)

4.2. User-Defined Reasoning

A more flexible reasoning mechanism is user-defined reasoning. Through the creation of user-defined reasoning rules within the entailment of first-order logic, a wide range of higher-level, conceptual context such as "what the user is doing" can be deduced from relevant low-level context. Table 3 shows the user-defined context reasoning rules that are employed to derive user's situation in the smart phone scenario.

5. Prototype Implementation

In this section, we will present results of our preliminary experiments with context reasoning. The objectives of these experiments are to conduct a quantitative feasibility study for logic reasoning in pervasive computing environments, and provide useful information for the implementation of context reasoning.

We used our prototype implementation of two context reaonsers (description logic based ontology reasoner and first-order logic based situation reasoner) to carry out experiments. Context reasoners are built using Jena2 Semantic Web Toolkit [7], which supports rule-based inference over OWL/RDF graphs. To synthesize large-scale context dataset, we have merged CONON and CYC Upper Ontology [8] to create several context datasets ranging from small-scale (about 1K RDF triples) to large-scale (more than 10K RDF triples). The size of the dataset is measured in term of the number of RDF triples, each of which represents a single S-V-O predicate. For example, a datasets containing 2534 OWL classes and instances is parsed into 10234 RDF triples. Current version of CONON containing 197 OWL classes (or 790 triples) can be seen as a small-scale context dataset, while CYC Upper Ontology containing 2885 classes (or 18777 triples) is a large-scale context dataset. The experiments have been conducted on a set of Linux workstations with different hardware configurations (512 MB RAM with P3/600 MHz, P3/1.2 GHz, and P4/2.4 GHz). The ontology reasoner we have tested is associated with the DL rule set consisting of all 111 axioms entailed by OWL-Lite, and the situation reasoner applies a rule set containing of 10 forward-chaining rules that we have partially described in table 2.

Figure 4 shows the results of the experiments. It is not surprising to see that the run time performance of logic-based context reasoning depends on three major factors: size of context information, complexity of reasoning rules, and CPU speed. The difference of performance between different datasets shows that context reasoning based on logics is a computational-intensive task. However, reasoning under current CPU speed is still feasible for non-time-critical applications.

For example, the real-time requirement for the smart phone service is not likely to be critical so that the delay of context reasoning (several seconds) is acceptable. The results also shows that run time of context reasoning largely depends on the complexity of rule sets. The user-defined reasoner using small rule set greatly outperforms the OWL reasoner with a large DL rule set over identical context datasets.

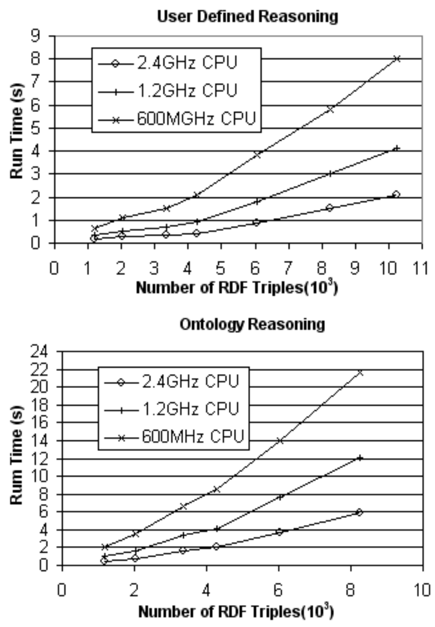

Figure 4. Run time performance of context reasoning.

From the quantitative study of runtime performance, we have a number of observations that are useful for the design of context model and context reasoning mechanism:

First, context reasoning is generally feasible for non-time-critical applications. For time-critical applications such as security and navigating systems, we need to control the scale of context dataset and the complexity of rule set. A tentative solution is to perform static, complex reasoning tasks (e.g., description logic reasoning for checking inconsistency) in an off-line manner.

Second, from system deployment point of view, we need to de-couple context processing and context usage in order to achieve satisfactory performance. In this way, context reasoning is independently performed by resource-rich devices such as a residential gateway; ubiquitous services hosted by thin clients can acquire high-level context from a centralized service, instead of perform excessive computation themselves.

Finally, the design of context model should take account of scalability issue. Context aware services in different domains shares most general concepts, while there exists significant difference between the ontologies they need. Hence, a scalable context model should be able to separate domain-specific ontologies for different system environments. The design of upper-level and domain-specific ontologies would take a promising step to control the scale of context dataset.

6. Conclusion

Our study in this paper shows that ontology based context model is feasible and necessary for supporting context modeling and reasoning in pervasive computing environments. We have implemented the CONON and logic based context reasoning schemes. In addition, we have conducted a performance study to evaluate the feasibility for context reasoning in pervasive computing environments. The work of this paper is a part of our ongoing context aware service infrastructure [9], which aims to provide an open, reusable infrastructure for essential context aware mechanisms. In particular, our design explores Web Ontology Language for context modeling and knowledge sharing, hybrid reasoning and learning for context interpretation, and Semantic Web query for expressive context query and resource discovery.

References

[1] A.K.Dey, et al. "A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications", Human-Computer Interaction Journal, Vol. 16(2-4), pp. 97-166, 2001.
[2] Tim Kindberg, et al. "People, Places, Things: Web Presence for The Real World", Technical Report HPL-2000-16, HP Labs, 2000.
[3] Karen Henricksen, et al."Modeling Context Information in Pervasive Computing Systems", Pervasive 2002.
[4] Anand Ranganathan, et al. "A Middleware for Context-Aware Agents in Ubiquitous Computing Environments", USENIX International Middleware Conference, 2002.
[5] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web", Scientific American may 2001.
[6] F. van Harmelen, et al. "Owl Web Ontology Language Reference", http://www.w3.org/TR/owl-ref/, 2002.
[7] Jena2 Semantic Web Toolkit: http://www.hpl.hp.com/semweb/jena2.htm.
[8] CYC Upper Ontology: http://www.cyc.com/cycdoc/vocab/vocab-toc.html.
[9] Daqing Zhang, Xiaohang Wang, Karianto Leman, and Weimin Huang, "OSGi Based Service Infrastructure for Context Aware Connected Homes", In 1st International Conference on Smart Homes and Health Telematics, 2003, France.