# CACOnt: A Ontology-Based Model for Context Modeling and Reasoning

Nan Xu, Weishi Zhang, Huadong Yang, Xiuguo Zhang, Xing Xing
School of Information Science and Technology
Dalian Maritime University
Dalian, China, 116026
{xunan, teesiv, yanghd8608, zhangxg, xingxing}@dlmu.edu.cn

***Abstract*—In this paper, we present a general and extensible context-aware computing ontology (CACOnt) for modeling context and providing inference mechanisms. CACOnt provides not only the generic context ontologies for capturing basic concepts about context, but also the extensibility for adding domain-specific ontologies in a hierarchical manner. CACOnt facilitates the context reasoning capabilities by providing semantic logics which is possible to combine with rule-based systems. However, the set of rules cannot entirely cover the domain of contexts, we present a semantic similarity-based rule matching algorithm as the solution to this problem.**

***Keywords-context modeling; context reasoning; semantic similarity; context-aware computing; rule matching***

## I. INTRODUCTION

Recently, people are increasingly interesting in context-aware computing systems which can not only proactively adapt their behaviors to the user's current situation, but also protect them from being disturbed with various kinds of devices and services while on their regular duty.

To realize an context-aware computing systems, it is very important that various kinds of information from diverse and heterogeneous of sources should be pulled together to form a representation model which must be agree on shared by all participating devices to support interoperability. Additional, context-aware computing systems should also perform reasoning over contexts which can guarantee the quality of the context, deduce implicit information and pass decisions about the actions to be triggered.

Ontology potentially provides a well-founded mechanism for the representation and reasoning of context information. In the context-aware computing environments, ontology is referred as the shared understanding of some domains, which is general considered as a set of entities, relations, functions, axioms and instances [1]. The use of ontologies brings us several benefits and additional functionalities for developing context models based on ontology: Formal knowledge represents, logic reasoning, knowledge sharing and reuse.

In order to provide formal semantics and efficient reasoning, the Web Ontology Language (OWL) which become the recommendation by W3C was created. OWL extends RDF and RDF by including more expressive constructors to describe the semantics of the elements. It provide mechanism to achieve the balance between expressiveness and computability consequently enable a formal knowledge representation that enhance the capabilities of model computational processing, its adaptability, and even promote their massive use [2]. Section 3 focuses on the proposed model defined by OWL.

The rest of the paper is structured as follows. Section 2 gives an overview of related work. In section 3 we propose our formal context model based on ontology. Section 4 shows how context reasoning can be used to enhance context-awareness. The rule matching algorithm and a corresponding case study is given in section 5. Section 6 summarizes this paper.

## II. RELATED WORK

Context-aware computing has been introduced as a key characteristic in many different domains over the last decade. Much research has been done in the area of context-aware computing that demonstrates the importance of context awareness.

Strang et al. [3] present a survey on context modeling approaches and gave a comparison among them: Key-value modeling which is the application-oriented approach lacking of the formal basis and does not support knowledge sharing across different systems. Markup scheme modeling approach such as CSCP, is difficult and non-intuitive to capture complex contextual relationships and constraints. Although object-oriented modeling, graphical modeling and logic-based modeling approaches support formality and some of them capture temporal aspect of context information, they do not address knowledge sharing and context reasoning issues. While, ontology-oriented modeling approach focuses on context ontology and explores the potential capability of context reasoning based on Semantic Web technologies.

Chen et al. [4] defined a context ontology based on OWL in their Context Broker Architecture (CoBrA), which only covers contexts in campus space, while has no explicit support for modeling general contexts in heterogeneous environments. And the reasoning capabilities based on Description Logic (DL) does not combined with rule-based reasoning. They also described SOUPA [5] in a radiating manner into SOUPA core and extension. SOUPA provides rich semantics for programming. It targets at to be a more general one that combines many useful vocabularies from different consensus ontologies.

Gu et al. [6] presented their context ontologies called CONON which organize their upper ontology and lower domain-specific ontologies into a tree hierarchy. While the design has been done with particular applications in mind, that is, their smart home and is not flexibly extensible beyond it. Using CONON, two types of contextual reasoning tasks are supported: ontology reasoning and user-defined

Published by Atlantis Press, Paris, France.

reasoning by defining specific rules in first-order logic. However, description language of rules is more complicated.

At present, in the aspect of rule matching, most approaches use rule engine such as Jena and Jess which both based on Rete algorithm that only supports exactly matching but not approximate matching. Liu et al. [7] applies semantic distance for the approximate rule matching, however, only hierarchical relations is considered.

## III. ONTOLOGY-BASED CONTEXT MODEL

Context information has a great variety. In realistic context-aware computing environments, it is usually grouped into a number of sub-domains for different intelligent environments such as home domain, office domain, and campus domain. Context in each domain shares common concepts that can be modeled using a general context model, while differs significantly in detailed features. Therefore, we adopt a two-layer hierarchical approach for designing our context ontology CACOnt. It is distributed into the generic context ontologies for the general concepts and the domain-specific context ontologies which apply to different sub-domains. This separation encourages the reuse of general concepts, reduces the scale of context knowledge, and also releases the burden of context processing [8].

As the evolving nature of context, it likely to be an insurmountable task to completely formalizing all context information of the intelligent environments. However, we can find the most fundamental elements of *Context Entity* including *User*, *Space*, *Environment*, *Device* and *Service* for capturing the general context information.

### A. User Model

According to Dey [9], context information is only relevant if it influences a user's task. This is why users play an important role in context-aware computing environments.

As shown in Fig. 1, our *User* model which is a subclass of the *Context Entity* describes the follow issues: The definition of the characteristics of users. The *Personal Profile* is used to describe user's characteristics such as name, identity, homepage and preference which provides by FOAF ontology. A *Schedule* consists of one or more events, each *Event* includes one or more tasks, and each *Task* includes one or more activities; the *Situation* represent what the user is doing or the current state of the users such as the roles, the current tasks, the accompanists and environments.

It links to *Environment* model for being provided *Physical Conditions* and relates *Device* model with '*owns*' property.

### B. Device Model

As simplified graphical representation shown in Fig. 2, this model which is also a subclass of *Context Entity* classifies *Device* into two types: *Computational* and *Non-computation* entity. Each of the devices has the *Hardware* and/or *Software Profile* which describes the general and particular characteristics.

Software that is available on the device can be described by the required parameters or properties such as the name, edition and version in the *Software Profile*. We distinguish four types of *Hardware Profile* that should be described in the context to support service providing or software deployment. Each of them has some aspects and properties that are important for subsequent performance measure.

It links to *Environment* model for being provided *Physical Conditions* and relates *Service* model with '*provides*' property.

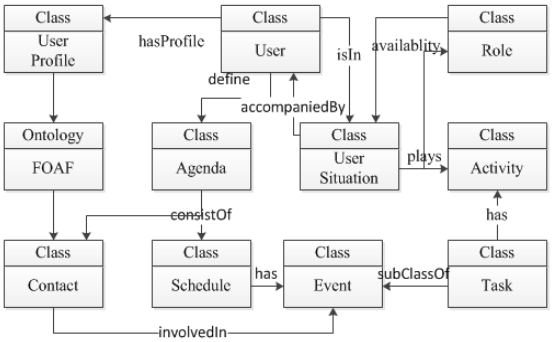


Figure 1. User Model

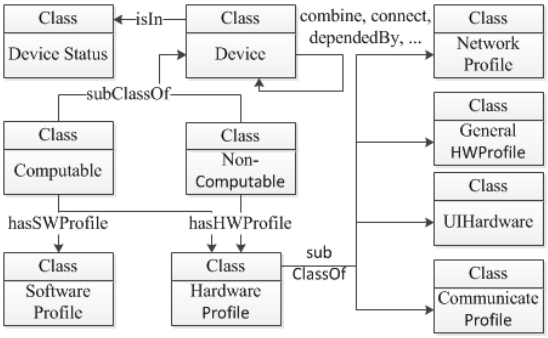


Figure 2. Device Model

### C. Sevice Model

While guarantying certain QoS aspects in mind, services should have the capacities of sensing and adapting with the current context. By referencing a service ontology called OWL-S, our *Service* model has a multi-level description.

*Service Profile*: It mainly provides a description of what is accomplished by the service, i.e. the functionality of the service by specifying its inputs, outputs, precondition, effect information, and non-functionality, i.e. quality of service.

*Service Model*: It describes how to ask for the service and what happens when the service is carried out by offering more detailed information about the control-flow and data-flow involved in using the service so that the user or agent could make a decision of whether the service meets its needs.

*Service Grounding*: It specifies the details of how to access a service. And it deals with implementation details by specifying a communication protocol, message formats and other service specific details.

An overview of this model is shown in Fig. 3.

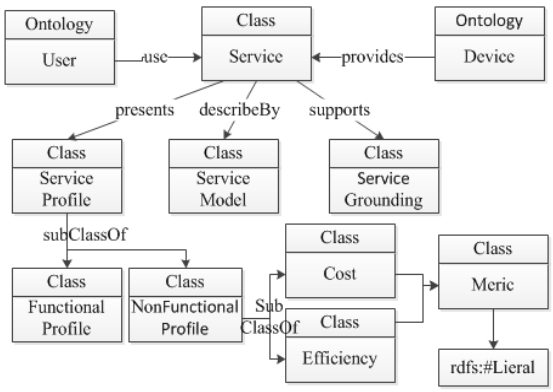


Figure 3. Service Model

### D. Space Model

Location information is the earliest form of context that the researchers noticed. Consequently, spatial information such as the locations of users, the spatial topology relationships between the buildings is undoubtedly the most important context information and must be addressed.

As the simplified graphical representation shown in Fig. 4, our *Space* ontology describes location information mainly from three aspects: *Geographical Location*, *Spatial Region* and *Political Entity*. The *Geographical Location* corresponds to *Coordinate* which represents through *latitude*, *longitude* and *altitude*. The *Spatial Region* defines symbolic representations of *Space* and typically represents geographical regions which is controlled by *Political Entity* that can relate to each other through the '*hasRCCRelationWith*' property which is based on RCC8. Furthermore, we define the *Spatial Region* as the union of the *Fixed* including *Indoor* and *Outdoor* regions and *Vehicle* structure. All the other context entities relate the Space model with '*locatedIn*' property.

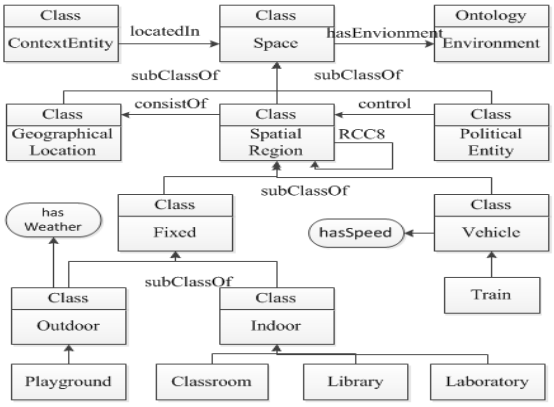


Figure 4. Space Model

### E. Environment Model

Environment continuously offers information that allows users to make appropriated decisions or that can influence their behaviors.

Our *Environment* model has been designed with a low level detail for adapting the model to the peculiarities of each environment. As illustrated in Fig. 5, it is related to the rest of context model through the '*hasEnvrionment*' property and mainly divided into three parts: *Physical Condition*, *Weather* and *Inanimate Object*. For the *Weather* part, we introduce a *Weather* ontology which is described in [10].

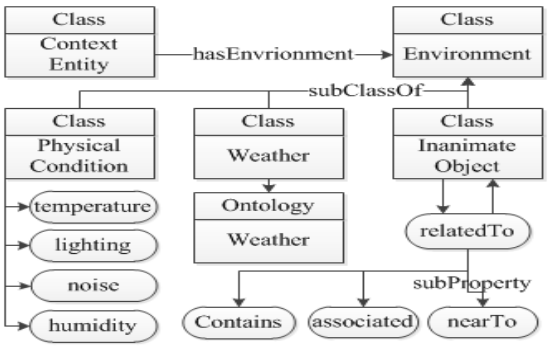


Figure 5. Envrionment Model

## IV. Context Reasoning

Context-aware systems must be able to perform context reasoning to facilitate dynamic adaptation to the changing environment, i.e. to be context-aware.

CACOnt which is a OWL encoded model provides inference capabilities based on DL which can combine with rule-based systems to improve reasoning capabilities [11].

By means of reasoning, context inconsistency can be checked, implicit information can be obtained, and the proactive behavior of particular services can be provided.

### A. Checking Consistency

On the basis of our context ontologies, the inconsistency is verified through rule-based reasoning provided by Jena engine [12]. Parts of the rules are shown in Table Ⅰ. For example, with the first piece of the rule, *a* is the subclass of *b*, and *b* is the subclass of *c*, then we can say *a* is subclass of *c*. However, if *a* is defined '*disjoinWith*' *b*, then there are conflict in the definition of class *a*, *b* and *c*.

TABLE I. PARTS OF RULES FOR INCONSISTENCY CHECKING

| Name | Reasoning Rules |
|---|---|
| SubClass Rule | (?a rdfs:subClassOf ?b), (?b rdfs:subClassOf ?c)-> (?a rdfs:subClassOf ?c) |
| DisjointWith Rule | (?X owl: disjointWith ?Y), (?A rdf:type ?X), (?B rdf:type ?Y)-> (?A owl: differentFrom ?B) |
| Inverse Rule | (?P owl:inverseOf ?Q), (?X ?P ?Y)-> (?Y ?Q?X) |

### B. Deducing Implicit Information

In this section, rules for driving high-level, implicit contexts include entailment rules and user-defined rules [13]. The sub-set rules are shown in Table Ⅱ. For example, with the *Together Rule*, if Mary is '*locatedIn*' office401, John is '*locatedIn*' room401, and office401 is the '*sameAs*' room401, then we can conclude that Mary is '*togetherWith*' John.

TABLE II. PARTS OF RULES FOR IMPLICIT INFORMATION DEDUCING

| Name | Reasoning Rules |
|---|---|
| Transitive Rule | (?P rdf:type owl:transitiveProperty), (?A ?P ?B), (?B ?P ?C) -> (?A ?P ?C) |
| SameAs Rule | (?A owl:sameAs ?B) -> (?B owl:sameAs ?A) |
| Together Rule | (?user1 CACOnt:locatedIn ?roomN), (?user2 CACOnt:locatedIn ?roomN) -> (?user1 CACOnt:togetherWith ?user2) |

### C. Providing proactive behavior of particular services

Based on the reasoning rules that are defined by the user himself or the developer, context-aware systems can proactively provide context-aware services. For example, according to the context of user such as location, speed, temperature and noise level of the environment, different reading manners can be provided dynamically. Table III shows parts of example rules.

TABLE III. PARTS OF RULES FOR SERVICE PROVIDING

| Name | Reasoning Rules |
|---|---|
| Train Rule | (?x cacont:locateIn ?train), (?x cacont:owns ?laptop), (?train cacont:hasSpeed ?50), (?train hasNoise ?5), (?train hasTemperature ?10) -> (?laptop provides ?picturesform) |
| Playground Rule | (?x cacont:locateIn ?Playground), (?x cacont:owns ?phone), (?x cacont:hasSpeed ?2), (?Playground hasNoise ?3), (?Playground hasTemperature ?5) -> (?phone provides ?audioform) |
| Library Rule | (?x cacont:locateIn ?library) , (?x cacont:owns ?PDA), (?x cacont:hasSpeed ?0) , (?library hasNoise ?1), (?library hasTemperature ?20) ->(?PDA provides ?wordsform) |

## V. SEMANTIC SIMILARITY-BASED RULE MATCHING

### A. Principal of Rule Matching Algorithm

Due to the variety nature of the context in the context-aware computing environment, the set of rules cannot entirely cover the domain of contexts. The rule that exactly matches current context information probably does not exist. We present a semantic similarity-based rule matching algorithm as the solution to this problem. In our method, based on ontology structure, we consider not only the hierarchical concepts but also the non-hierarchical binary relations for estimating the instance similarity.

*1)* If the range of the kind of context is numeric, the similarity is computed as:

$$sim(i_1, i_2) = 1 - \frac{|v(i_1) - v(i_2)|}{range} \quad (1)$$

Where *v(i)* represents the value of instant *i* on this kind of context, range represents the range of value for this kind of context. While, it can be also applied to the context which range is boolean or string through a transformation.

*2)* If the range of the kind of context is a semantic concept, we introduce semantic similarity for characterizing the similarity between the two concepts:

*a)* We introduce *depth* to denote the number of edges on the path from it to the root node [14]. $LC(i_1, i_2)$ denote the lowest common concept node of both $i_1$ and $i_2$. The similarity is calculated as follows:

$$sim_r(i_1, i_2) = \frac{2 * depth(LC(i_1, i_2))}{depth(i_1) + depth(i_2)} \quad (2)$$

*b)* While depth-based similarity measure takes only the inherited relations but not the binary relations into account. Here, we introduce a function *F(c)* which can return a set of the properties of the node *i*; The binary relation similarity is calculated as follows:

$$sim_b(i_1, i_2) = \frac{2*|F(i_1) \cap F(i_2)|}{|F(i_1) \cap F(i_2)| + |F(i_1) \cup F(i_2)|} \quad (3)$$

Given two contextual concept we can calculate their semantic similarity that is a global similarity measure aggregating the Equation (2) and (3).

$$sim(i_1, i_2) = \frac{1}{2} sim_i(i_1, i_2) + \frac{1}{2} sim_b(i_1, i_2) \quad (4)$$

Due to context-aware rules may involve a variety kinds of contexts, it necessary to evaluate the similarities of the each components of rule's predecessor with corresponding current context, and then the weighted average of these similarities can be consider as the similarity degree between the current context and the predecessor of a certain rule that defined in Equation (5).

$$Similarity(CV, R) = \sum_{i=1}^{n} W_{C_i} * sim(V_{C_i}, R_{C_i}) \quad (5)$$

Where $W_{Ci}$ is an experience parameter that present the weight of context $C_i$; $V_{Ci}$ and $R_{Ci}$ stand for the value of a kind of current context and the predecessor of a rule upon $C_i$ respectively and $Similarity(V_{Ci}, R_{Ci})$ is the similarity between $V_{Ci}$ and $R_{Ci}$.

**Algorithm 1: Rule matching algorithm**

**Input:** CACOnt: our context ontologies, RuleSet: a set of predefined rules, CV: current context, *W*: weight

```
01: max_sam = 0
02: for each rule in RuleSet do
03:    Similarity(CV,R) = 0
04:    for each component of rule's predecessor do
05:    Compute Similarity(CV,R)
                    according to Equation (1) - (5)
06:    end for
07:    if (Similarity(CV,R) > max_sim) then
08:       Matched Rule = Rules[i]
09:    end if
10: end for
```

**Output:** the matching rule *Matched Rule*

The pseudo code describes see *Algorithm 1*, its time complexity is O(*rc*), where *r* is the number of rules, *c* is the number of the kinds of context that involve in the rule which holds the maximum number.

### B. A Case Study

In this section, we further illustrate the principle of the rule matching algorithm through an example based on our context model CACOnt.

Assume that Mary is sitting in classroom with a PDA in her hand for reading something through WiFi, and the value of temperature and noise level is 18 and 1 respectively. Obviously, the rules which are shown in table Ⅲ that exactly matches current context information does not exist. So it is necessary to choose a rule that most close to the current context upon the semantic.

According to the Fig. 4, the inherited relation-based similarity between concept classroom and train, playground, library in turn is 0.29, 0.5, 0.75 through Equation (2). By means of Equation (3), binary relation-based similarity between them is 0.89, 0.89, 1 respectively. Then the global similarity is 0.59, 0.695, 0.875. Analogously, due to PDA, laptop and phone are all of subclasses of computational entities, the global similarity between them is the same which is 0.75.

We apply the Equation (1) on the measure of the speed, temperature and noise level and obtain the similarities for the Train Rule are 0.96, 0.87, 0.5, for the Playground Rule are 0.98, 0.78, 0.98 and for the Library Rule are 1, 0.97, 1.

Here we set the each weight is 0.2 for convenient. Therefore, the degree of similarity between the current context and the rules are obtained that are 0.734, 0.837, 0.969 respectively. Obviously the Library Rule can be considered as the matching result because of its predecessor is the most close to the current context.

## VI. CONCLUSION

In this paper we present an ontology-based general and extensible context model in a hierarchical manner that includes the generic ontologies and the domain-specific ontologies. Based on our context model, a hybrid approach of context reasoning based on ontology and rules is provided. Moreover, the rule that exactly matches current context probably does not exist. To solve this problem, we present a semantic similarity-based rule matching algorithm in which both inherited relations and binary relations are considered. In addition, a case study is also explained.

## ACKNOWLEDGMENT

This research is supported by the National Science Foundation of China (Grant No.60973013, Grant No.61272172) and the Fundamental Research Funds for the Central Universities in China (Grant No.2011QN027).

## REFERENCES

[1] J. Ye, L. Coyle, S. Dobson and P. Nixon, “Ontology-based models in pervasive computing systems,” The Knowledge Engineering Review, vol. 22, Dec. 2007, pp. 315-347, doi:10.1017/S0269888907001208.

[2] R. Hervás and J. Bravo, “COIVA: context-aware and ontology-powered information visualization architecture,” Software: Practice & Experience, vol. 41, Apr. 2011, pp. 403-426, doi: 10.1002/spe.1011.

[3] T. Strang and C. Linnhoff-Popien, “A context modeling survey,” Proc. 1st International Workshop on Advanced Context Modelling, Reasoning and Management, in conjunction with 6th International Conference on UbiComp, Sep. 2004.

[4] X. H. Wang, D. Q. Zhang, T. Gu and H. K. Pung, “Ontology based context modeling and reasoning using OWL,” Proc. Pervasive Computing and Communications Workshops, IEEE Press, Mar. 2004, pp. 18-22, doi:10.1109/PERCOMW.2004.1276898.

[5] H. Chen, T. W. Finin and A. Joshi, “The SOUPA ontology for pervasive computing,” Ontologies for agents: Theory and experiences, Aug. 2005, pp. 233-258, doi:10.1007/3-7643-7361-X_10.

[6] T. Gu, S. Chen, X. Tao, J. Lu, “An unsupervised approach to activity recognition and segmentation based on object-use fingerprints,” Data and KnowledgeEngineering, vol. 69, Jun. 2010, pp. 533-544, doi:10.1016/j.datak.2010.01.004.

[7] D. Liu, X. W. Meng, J. L. Chen and Y. M. Xia, “Algorithms for rule generation and matchmaking in context-aware system,” Journal of Software, vol. 20, Oct. 2009, pp. 2655-2666, doi: 10.3724/SP.J.1001.2009.03436.

[8] C. Bettini, O. Brdiczka, K. Henricksen, J. Indulska, D. Nicklas, A. Ranganathan and D. Riboni, “A survey of context modelling and reasoning techniques,” Pervasive and Mobile Computing, vol. 6, April. 2010, pp. 161-180, doi: 10.1016/j.pmcj.2009.06.002.

[9] A. K. Dey, “Understanding and using context,” Personal and Ubiquitous Computing, vol. 5, Feb. 2001, pp. 4-7, doi:10.1007/s007790170019.

[10] http://mnemosyne.umd.edu/~aelkiss/weather-ont.daml.

[11] M. Sensoy, W. Vasconcelos and T. Norman, “Combining semantic web and logic programming for agent reasoning,” Advanced Agent Technology, vol. 7068, Oct. 2012, pp. 428-441, doi:10.1007/978-3-642-27216-5_33.

[12] Jena: A Semantic Web Framework for Java. 2006. http://jena.sourceforge.net/index.html.

[13] P Chen, S Sen, HK Pung, W Xue and W. Wong, “A context management framework for context-aware applications in mobile spaces,” International Journal of Pervasive Computing and Communications, vol. 8, Feb. 2012, pp. 185-210, doi:10.3724/SP.J.1001.2009.03436.

[14] K. Zhang, J. Tang, M. Hong, J. Li and W. Wei, “Weighted ontology-based search exploiting semantic similarity,” Lecture Notes In Computer Science, vol. 3841, Jan. 2006, pp. 498-510, doi:10.1007/11610113_44.