Modeling and Reasoning about Uncertainty in Context-Aware Systems*

Binh An Truong, Young-Koo Lee and Sung-Young Lee
Department of Computer Engineering, Kyung Hee University
Giheung-Eup, Yongin-Si, Gyeonggi-Do, 449-701, Korea
tabinh@oslab.khu.ac.k, {yklee,sylee}@khu.ac.kr

Abstract

Uncertainty always exists as an unavoidable factor when developing context-aware applications for pervasive computing environments [1][2][3][4]. In this paper, we propose a unified context model to support representation and reasoning about uncertain context. Our unified context model extends the existing, de-facto ontology-based context models with probabilistic models to support probabilistic reasoning. Especially, our context model can be easily integrated with existing ontology-based context-aware systems. Given the unified context model, unified context ontology can be built and used as frameworks in developing context aware applications. Besides, our recipe of supporting the probabilistic reasoning is flexible and adaptive to the highly variable features of pervasive computing environments.

1 Introduction

Pervasive computing is a vision in which computing systems are seamlessly integrated into the everyday lives of users. Context-awareness is the key to enable that vision [5]. Context aware applications can provide relevant services and information according to the user's context. Context aware systems can react and adapt to changes of its current surroundings. Context might be considered as a collection of information which characterizes the interaction between a user and the application. Time, location, temperature, lighting, sound and activity are examples of contextual information or context.

Uncertainty is an unavoidable factor in any context-aware system. It is mostly caused by the imperfectness and incompleteness of sensed data. As a result, the high-level information deduced from raw sensing data may not be accurate. For example, it is difficult to deduce that the user is sleeping based on low-level sensing information such as his location is in bed; the room light is dark and the sound is quiet. In addition, the underlying logical and rule-based reasoning mechanisms of existing context aware systems do not support reason about uncertainty. Hence, to be more reliable and adaptive, context-aware systems must have the capability of modeling as well as reasoning about uncertainty.

The Bayesian networks (or probabilistic models) method, which is used commonly in AI community, is a very powerful method for the representation and reasoning about the uncertainty. A Bayesian network is a representation of a full joint distribution over a set of random variables. It can answer queries about any of its variables given any evidence. Besides, Bayesian network provides different forms of reasoning including: prediction (inferring results from causes), abduction (inferring causes from results) and finally, explaining away (the evidence of one cause reduces the possibility of another cause given the evidence of their result) which is especially difficult to model in logical rule-based systems [6]. Nevertheless, a fundamental limitation of using Bayesian network for knowledge representation is that it can not represent the structural and relational information. Also, the applicability of a Bayesian network is largely limited to the situation which is encoded, in advance, using a set of fixed variables. Thus, it is not suitable for representation of contextual information which is highly interrelated in pervasive computing environments [7].

In this paper, we propose an integrated modeling approach which inherits the advantages from both Bayesian network and ontology. Our unified context model, which is influential by the Probabilistic Relation Model (PRM) [8], can be considered as the glue for integrating the probabilistic models into ontology. Given the unified context model, we can build a unified context ontology which captures both structural and relational knowledge as well as the probabilistic knowledge of a domain. The unified

* This research was supported by the MIC (Ministry of Information and Communication), Republic of Korea, under the ITRC (Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Assessment).

context ontology specifies two parts of knowledge in a domain: (1) the relational schema which defines concepts and relations in term of classes, properties, relation, constraint; and (2) probabilistic models which specify the conditional probabilistic dependencies between concepts in that domain. Given the unified context ontology, Bayesian networks, which is used to reason about uncertain context, are derived in an autonomous and adaptive fashion which reflects most truthfully about the current state of the domain.

The rest of paper is organized as follows. Section 2 describes a scenario which will be used through out the paper as an example for illustrating of our approach. In section 3, we introduce the concept of our novel context model. Section 4 describes an ontology which is used to represent the uncertain contexts in domain-oriented ontologies. Section 5 introduces three types of reasoning supported given the context ontology defined based on our context model. Section 6 mentions related works and makes some comparisons between our work and previous works. Finally, the paper end with conclusion in section 7

2 A Smart Home Scenario

Our smart automated home can proactively control the environmental conditions to reduce resource consumption. The smart home system follows some algorithms to minimize the energy consumption while still making the inhabitants comfortable. The windows and blinds can be controlled automatically according to the situations to provide optimal cooling or heating strategy or to create a fresh air breeze.

For instance, on a day when the temperature is shifted from cool to warm, the smart home system might determine that the optimal warming strategy is to open the windows and blinds so that the warm air can go inside. This scenario seems to be very simple but it is practically more complicated in the real situation. There are situations that the system needs to consider in order not to fail in "do the right thing", for example: the outside is so noisy while there are people reading inside the room; someone does not want the blinds open because he/she is sleeping; the air outside is polluted by dust and smoke, and so on. The decision to open or close the windows and blinds depends on many elements in the situation. Such dependencies are also changed from situation to situation.

In the next section, we will show how to model this smart-home domain based upon our proposed context model.

3 The Unified Context Model

Our context model is influenced mainly by the Probabilistic Relational Model developed by Friedman et al. in [8] and the Probabilistic Frame-based systems of Koller et al. in [9]. We made several modifications to make our context model simple and suitable to our requirements.

Our context model consists of two parts:

- Relational schema which represents the structural and organizational information in forms of class, binary relations, relation chains and properties.
- Probabilistic models which annotate the probabilistic dependence relationship between properties of classes.

3.1. The Relational Schema

The basic unit of our context model is a class X. A class may be a sub class of another (its super class). A class includes a set of *relations* $R_1,\ldots,R_n$ and a set of *properties* $P_1,\ldots,P_n$. A relation R_i specifies a binary relationship between two classes X and Y. All relations are typed appropriately. The binary relationship $X.R(Y)$ can be considered as an object-property of class X which has the value-type of class Y. For example, in Figure 1, the binary relation *hasWindow* define the relationship between class *Room* and class *Window*.

A *relation-chain* is a sequence of binary relations separated by period. A relation-chain creates an implicit relational link between two classes in a relational domain. A relation-chain $X.R_1.R_2\ldots R_n$ refers to the final class which is the type of the final relation in the chain. Each relation R_i in the chain must be correctly typed. In Fig. 1, the relation-chain *ofWindow.ofRoom* is an indirect relation between class *WindowAgent* and class *Room*.

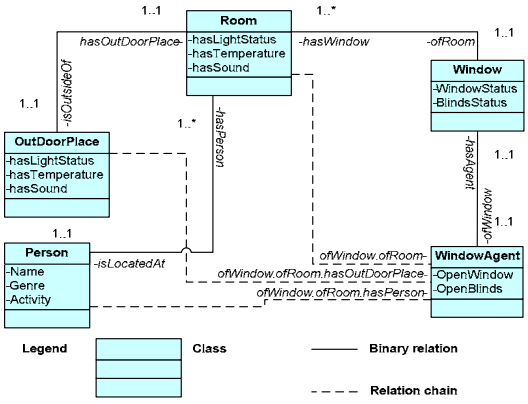

Figure 1: A relational schema of scenario

A property $X.P_i$ may have some associated *restrictions (or facet)*. A restriction can be simply a standard value-type which specifies a range of values for a property.

A *property-chain* is formed by appending a relation-chain with a property of the referenced class. It specifies a reference from a class to another class's property. For example, the property chain *ofWindow.ofRoom.hasLight-Status* is a reference link from class *WindowAgent* to the property *hasLightStatus* of the class *Room*.

3.2. The Probabilistic Model

We use probability for representing the uncertainty within a domain. A class which consists of probabilistic information is annotated with the local probabilistic model. This type of class is called p-class. A p-class, similarly to the normal class, has properties, relations and restrictions.

A property is either simple or complex. A p-class may also have some properties that do not participate in the probabilistic model, whose type is neither of the above. For example, the *Person* class may also have the property *Name*, which does not have an associated probabilistic model. This feature allows existing knowledge bases to be annotated with probabilistic information without requiring a complete redesign of the ontology.

A simple property corresponds with a root node in Bayesian network. A simple property has two restrictions: *hasValue* and *hasPD*. The restriction *hasValue* is an explicitly enumerated list of possible values for the property. The restriction *hasPD* specifies the probability distribution over the values listed in the *hasValue* restriction.

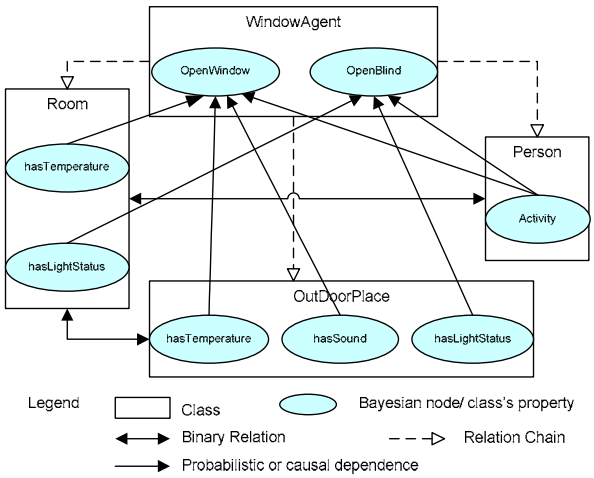

Figure 2: The probabilistic model for the scenario

For example, the property *Temperature* of the class *Room* may have the restriction *hasValue* as *{Hot, Warm, Cool, Cold}* and the restriction *has-PD* as *{0.3, 0.25, 0.15, 03}*. The sum of all values listed in a restriction *hasPD* must be equal to 1 to satisfy the probability axioms.

A complex property corresponds with a node in Bayesian network which has a set of parent nodes. Beside two restrictions: *hasValue* and *hasPD* defined above, a complex property has two other restrictions: *hasParents* and *hasCPT* which specify the conditional probabilistic dependencies on other properties. The *hasParents* restriction of the complex property *P* specifies a list of property-chains on which the value of this property depends. Each property-chain refers to one property of other class. For example, in the *WindowAgent* class, the parent of the property *OpenWindow* may be the property-chain *ofWindow.-ofRoom.hasTemperature*. The *hasCPT* restriction specifies the conditional probability distribution over the values of the property given values of its parents, which are listed in the hasParents restriction. The conditional probability distribution is specified using a conditional probability table (CPT) as in Bayesian networks. For each combination of values of its parents, the CPT provides a probability distribution over values of the property. For simplicity, we assume that the CPTs are represented as fully specified functions of parent values.

4 A Unified Context Ontology

4.1. A two-layer ontology

Based on the proposed context model, we can specify the ontology to capture the knowledge of a domain. The p-class can be used just like any other normal class. We can create instances of class, which inherit all of its template properties and restrictions. In particular, the probability distribution over values of properties of the instance will be as described in the p-class. Similarly, the inheritance mechanism can be used to make one p-class a subclass of another. A subclass can extend the definition of the super-class as well as overwrites parts of it. It can redefine the probability model of one or more of the properties.

The context ontology should be able to captures all the characteristics of context information. As the pervasive computing domain can be classified into a collection of sub-domains such as home-domain, office-domain, university-domain, market-domain, etc, it would be easy to specify the context in one domain in which a specific range of context is of interest. The separation of domains can also reduce the burden of context processing.

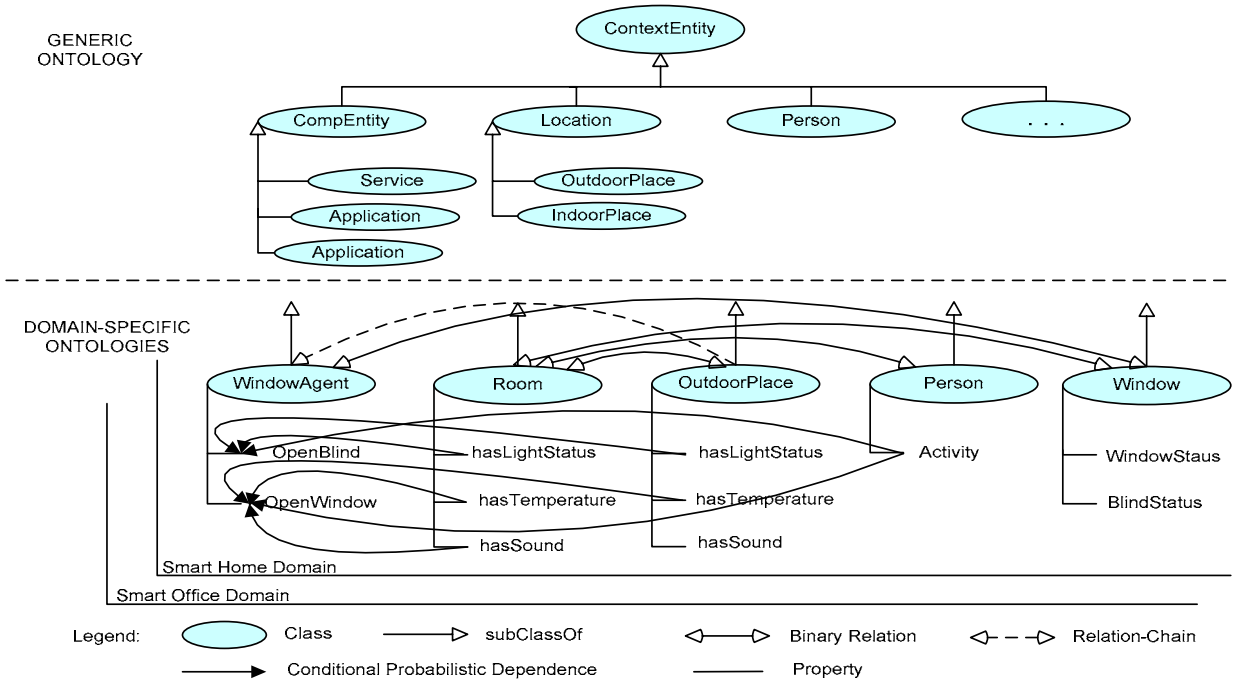

Figure 3 A two-layer context ontology for relational and probabilistic knowledge

Our context ontology is divided into two layers including a generic ontology layer and domain-specific ontologies layer as shown in Fig. 3:

- *The generic ontology* is a high level ontology which captures general context knowledge about physical world in pervasive computing environment.
- *The domain-specific ontologies* are a collection of low-level ontologies which define the details of concepts and properties in each sub-domain. A low-level ontology in each sub-domain consists of two parts: (1) relational schema which captures all relations, relation-chains of the sub-domain; and (2) probabilistic models which capture all conditional probabilistic dependencies between properties in that sub-domain.

The details of each generic concept, such as relations, relation-chains, conditional probabilistic dependences, are redefined in the domain-specific ontologies and may vary from one domain to another.

4.2. Ontology language: PROWL

We use Web Ontology Language (OWL) [10] for representing the context ontology. However, to represent new concepts such as relation-chains, property-chains and probabilistic-dependencies, we augment additional language elements. We call the derived language PROWL.

4.2.1 PROWL for relational schema

A relation of our context model is equivalent to the `owl:ObjectProperty`. We define a binary relationship as an `owl:ObjectProperty`.

We define a class `RelationChain` to represent the relation-chain information between two classes. Each relation-chain is represented by an instance of the `RelationChain` class with a string to represent the chain.

```
<RelationChain rdf:ID="RC-ofWindow.ofRoom">
    <hasRelationChain rdf:datatype="&xsd;string">
        ofWindow.ofRoom</hasRelationChain>
</RelationChain>
```

Figure 4: Class RC for relation chains

A property-chain is defined by a class `PropertyChain`. It includes a relation-chain instance and a string for specifying the name of property of the referred class by the relation-chain instance.

```
<PropertyChain rdf:ID="PC-
      ofWindow.ofRoom.hasLightStatus">
  <hasRelationChain rdf:resource="#RC-
      ofWindow.ofRoom"/>
  <hasProperty rdf:datatype="&xsd;string">
      hasLightStatus</hasProperty>
</PropertyChain>
```

Figure 5: A relation chain definition

4.2.2 PROWL for probabilistic relational model

As mentioned above, the OWL language does not support to describe new concepts that we use in our context model. Thus, we introduce three new additional elements: *rdf:hasDist, rdf:hasParents* and *rdf:hasCPT*. We define these language elements as a *rdf:List*

```
<rdf:Property rdf:ID="hasDist">
  <rdfs:label>hasDist</rdfs:label>
  <rdfs:domain rdf:resource="#Restriction"/>
  <rdfs:range rdf:resource="&rdf;List"/>
</rdf:Property>
```

Figure 6: Definition of a new property element

To define the probability distribution over values for a property, we use the *rdfs:range* together with *owl:oneOf* to specify an enumerated list of possible values for a property. To define the probability distribution over this list element, we use the additional elements *rdf:hasDist*, which can be applied to both data-type and object properties. Fig. 7 shows the probability distribution over the values of property *hasLightStatus* as *{{Bright 0.3} {Dark 0.4} {Dim 0.3}}*.

In Fig. 8, we define the conditional probabilistic dependencies of property *WindowAgent.-OpenBlind* with data-range *{Open, Close}* on two parent properties: (1) *Room.hasLightStatus* with data-range *{Bright, Dark}* and (2) *OutdoorPlace.hasLightStatus* with data-range *{Bright, Dark}*. The conditional probability distribution of *WindowAgent.-OpenBlind(Open)* is represented as a list *{0.3, 0.9, 0.6, 0.4}*.

```
<owl:ObjectProperty rdf:ID="hasLightStatus">
 <rdfs:domain rdf:resource="#Room"/>
  <rdfs:range> <owl:Class>
    <owl:oneOf rdf:parseType="Collection">
    <LightStatus rdf:ID="Bright"/>
    <LightStatus rdf:ID="Dark"/>
    <LightStatus rdf:ID="Dim"/>
  </owl:oneOf> </owl:Class>
 </rdfs:range> </owl:ObjectProperty>
<owl:Class rdf:ID="Room">
 <rdfs:subClassOf> <owl:Restriction>
   <owl:onProperty>
    <owl:ObjectProperty rdf:ID="hasLightStatus"/>
   </owl:onProperty> <rdf:hasDist> <rdf:List>
     <rdf:first rdf:datatype="&xsd;integer">
               0.3</rdf:first>
     <rdf:rest> <rdf:List>
      <rdf:first rdf:datatype="&xsd;integer">
                0.4</rdf:first>
      <rdf:rest> <rdf:List>
       <rdf:first rdf:datatype="&xsd;integer">
                 0.3</rdf:first>
       <rdf:rest rdf:resource="&rdf;nil" />
      </rdf:List> </rdf:rest>
     </rdf:List> </rdf:rest>
    </rdf:List> </rdf:hasDist>
  </owl:Restriction> </rdfs:subClassOf>
</owl:Class>
```

Figure 7: PROWL for probability distribution

```
<owl:Restriction>
  <owl:onProperty>
   <owl:ObjectProperty rdf:resource="#OpenBlind"/>
  </owl:onProperty>
  <rdf:hasParents>
   <rdf:List>
    <rdf:first rdf:resources="#PC-
ofWindow.ofRoom.hasLightStatus">/>
    <rdf:rest> <rdf:List>
      <rdf:first rdf:resources="#PC-
ofWindow.ofRoom.hasOutdoorPlace.hasLightStatus"/>
      <rdf:rest rdf:resource="&rdf;nil" />
    </rdf:List> </rdf:rest>
   </rdf:List>
  </rdf:hasParents>
  <rdf:hasCPT>
   <rdf:List>
    <rdf:first rdf:datatype="&xsd;integer">
              0.3</rdf:first><rdf:rest>
     <rdf:List>
      <rdf:first rdf:datatype="&xsd;integer">
                0.9</rdf:first><rdf:rest>
       <rdf:List>
        <rdf:first rdf:datatype="&xsd;integer">
                  0.6</rdf:first>
         <rdf:rest>
          <rdf:List>
           <rdf:first df:datatype="&xsd;integer">
                     0.4</rdf:first>
           <rdf:rest rdf:resource="&rdf;nil" />
          </rdf:List>
         </rdf:rest>
       </rdf:List> </rdf:rest>
     </rdf:List> </rdf:rest>
   </rdf:List>
  </rdf:hasCPT>
 </owl:Restriction>
```

Figure 8: Conditional Probabilistic Dependencies

5 Reasoning about context

Based on the context ontology that supports the representation of both ontological and probabilistic knowledge, we could construct a context knowledge base for the application domain. Reasoning about context information in the domain is supported by three types of reasoning mechanism: ontological reasoning, rule-based reasoning and Bayesian reasoning.

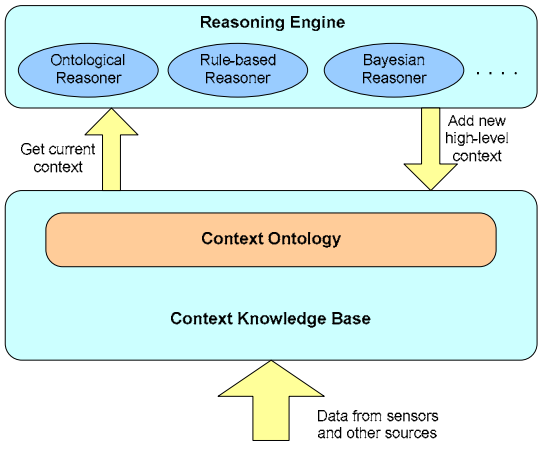

Figure 9: Three supported reasoning mechanisms

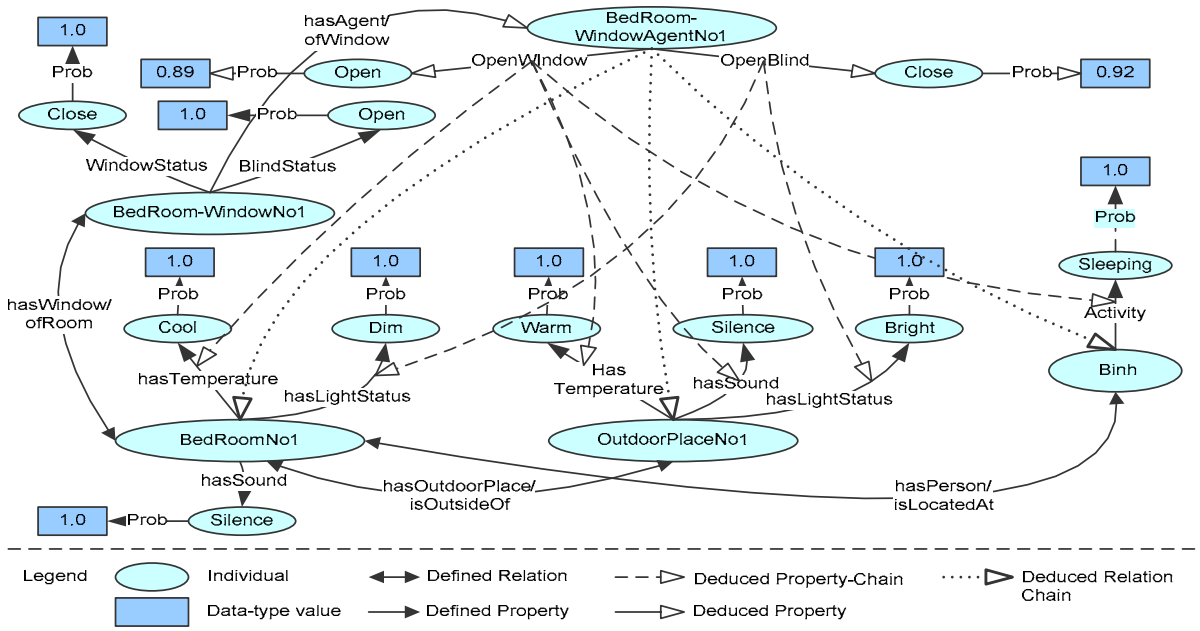

Figure 10: An example of the context ontology for the scenario in section 2

5.1. Rule-based reasoning

Rule-based reasoning mechanism is the default reasoning mechanism supported by the context ontology. However, when applying the rule to a simple or complex property of a p-class is that two update steps are executed. One updates the value of the property. The other updates the probability corresponding to that property's value as 1 and set other probability as 0 to satisfy the probability axioms.

For example, in our scenario, the context WindowAgent.OpenBlind can be deduced from the sensed context {OutdoorSpace.hasLightStatus, Room.hasLightStatus} as follows:

Prob(hasLightStatus(OutdoorSpace, Bright), 1.0) ^
Prob(hasLightStatus(Room, Dim), 1.0) ^
Prob(Activity(Binh, Sleeping) , 1.0)
$\Rightarrow$ *Prob(OpenBlind(WindowAgent, Open), 1.0)*

Our rule-based engine is developed based on the Jena Framework [11].

5.2. Ontological reasoning

The ontological reasoner can be described as an instance of the rule-based reasoner. It works by propagating implication, predefined rules over the instance data. For example, the relation-chain *ofWindow.ofRoom* of the instance *BedRoom-WindowAgentNo1* of the class *WindowAgent* will be an instance *BedRoom* of class *Room* so that the relation-chain *ofWindow.-ofRoom* satisfies.

5.3. Probabilistic reasoning

Before standard Bayesian inference can be used to answer queries about the values of the properties of the instances, a Bayesian network is derived from the context ontology. Depending on the domain, there may be more than one derived Bayesian network corresponding to each probabilistic relational model from the ontology.

The algorithm Construct-BN to derive a Bayesian network is described as follows. Each node in the Bayesian net B has the form I.P where I is an instance of a p-class and P is a property. The algorithm maintains a list L of nodes to be processed. Initially, L contains only the simple properties of named instances. In each iteration, the algorithm removes a node from L and processes it. The removed node I.P is processed as follows. For each parent I.RC.Pi, which refers to a property of another instance, an edge is added form I.RC.Pi to I.P; If I.RC.Pi is not already in B, we add I.RC.Pi into B and L; when all parents of I.P have been added, the CPT is constructed from the has-CPT restriction of I.P.

Since the Bayesian network is available, the standard Bayesian reasoner can use that network to infer about the probabilities of all nodes. Then, the probability of each node is updated directly to the property of instances the ontology. We implemented the Bayesian reasoner based on the API of Microsoft Belief Network software [16].

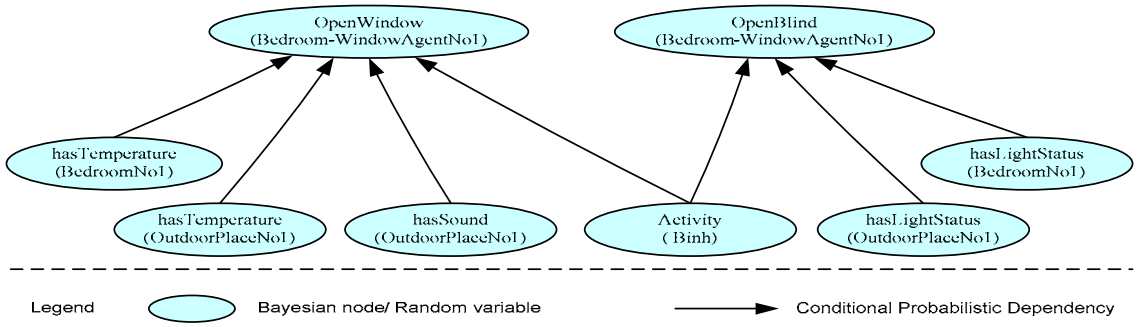

Figure 11: A derived Bayesian network given the context ontology in figure 10

Fig. 10 shows the current status of the context ontology for the scenario in section 2. We have evidences about the temperature, light status and sound of both bedroom and its outdoor place. We also have the information about the activity of the person (Binh). Hence, the probabilities of correspondent properties are set to 1. Given the context ontology, a Bayesian network is derived with structure shown in Fig. 11. After the Bayesian reasoning is finished, the inferred probabilities are updated to the context ontology as shown in Figure 10.

6 Related work

There has been some research in addressing the issue of uncertainty in context aware computing. First efforts tried to modeling the uncertainty of context information using various terms such as "imperfectness" [12], "confidence" [13], "accuracy" [14], etc. Nevertheless, those approaches lack of expressiveness to capture rich types of context information and they do not support the reasoning mechanism.

Using Bayesian networks to model and reason about the uncertain context was received much attention by context-aware research community recently. Some research has been used Bayesian network as the solution to reason about uncertainty. Ranganathan et al. [3] used Microsoft's Belief Network (MSBN) software to create the Bayesian networks structure. The Bayesian network is defined by knowledge expert and is mapped to predicates of instance in the ontology by developer. Tao Gu et. al. [15] represented directly a fixed Bayesian network over the properties of instances in the OWL-based ontology and then translates from the RDF graph into a Bayesian network for reasoning.

These two approaches solved the major problem of dealing with uncertainty by providing the probabilistic reasoning mechanism. However, their approaches to support uncertain reasoning are application-specific since the representation method does not represent the knowledge supporting reasoning. Even though the uncertainty is represented in ontology in [3], it still needs much help from knowledge expert and developers before it can be used by an appropriate reasoning mechanism. Whereas in [15], every new application needs help from knowledge experts to define the new Bayesian network in the ontology even if the domain knowledge is similar. Also, the mapping between a Bayesian network and the ontology is done manually by developers in both approaches. Even when the probabilistic knowledge in form of Bayesian networks is integrated into the context ontology in [15], it is still unable to reuse even for a similar application. The major reason is that the Bayesian network is defined over the instances which are distinctive for a typical application. In summary, both approaches provide no systematical method to integrate Bayesian reasoning mechanism into the ontology-based models.

The major characteristic in our approach is that we define the probabilistic information at the concepts level. We not only specify the uncertainty of concepts value (property's value) but also specify the probabilistic relationships between those concepts. Since ontology mainly deals with concepts within a domain, our context model can easily extend the current ontology-based modeling approach. Based on our unified context model, we can easily define a unified, domain-oriented context ontology which captures both logical or relational and probabilistic knowledge. Given that unified context ontology, we can build several knowledge bases for similar applications. For example, we can model a smart-home domain and build the smart-home ontology. For every new smart-home applications, we only need to specify the instances given that predefined smart-home ontology without redefine or construct a new one. Besides, we can add probabilistic information into an existing ontology by adding concepts such as relations, relation chains and conditional probabilistic dependencies without construct a brand-new context

ontology from the scratch. Thus, our work supports scalability and reusability with respect to knowledge modeling. Since the mapping-relations between nodes in Bayesian networks and properties of classes are explicitly defined in the ontology, the mapping process can be operated automatically without help from knowledge experts and developers. This feature reduces much burden on knowledge experts and developers in comparison with previous works [15], [3]. In addition, as probabilistic reasoning is supported, we can easily extend from reasoning to learning about uncertain context, which is simply learning about the parameters of Bayesian networks. The learning makes the Bayesian reasoning more robust and adaptive in highly variable pervasive computing environments.

7 Conclusions

This paper describes our approach of representing and reasoning about uncertain context. Our study in this paper shows that the proposed context model is feasible and necessary for supporting context modeling and reasoning in pervasive computing. Our work is part of an ongoing Context Aware Middleware for Ubiquitous System, which attempts to provide an easy, reusable infrastructure to develop ubiquitous context-aware applications. We are exploring method to integrate multiple reasoning methods from AI area and their supported representation mechanism into the context reasoning and management layer.

REFERENCES

[1] M. Satyanarayanan, "Coping with uncertainty", *IEEE Pervasive Computing*, page 2, Volume 2, Issue 3, July-Sept. 2003

[2] S. Benford, R. Anastasi, M. Flintham, C. Greenhalgh, N. Tandavanitj, M. Adams, J. Row-Farr, " Coping with uncertainty in a location-based game", *IEEE Pervasive Computing*, pp 34-41, Volume 2, Issue 3, July-Sept. 2003

[3] A. Ranganathan, J. Al-Muhtadi, R. H. Campbell, "Reasoning about Uncertain Contexts in Pervasive Computing Environments", *IEEE Pervasive Computing*, pp 62-70, Volume 3, Issue 2, Apr-June 2004.

[4] W. Abdelsalam, Y. Ebrahim, "Managing uncertainty: modeling users in location-tracking applications", *IEEE Pervasive Computing*, pp 60-65, Volume 3, Issue 3, July-Sept. 2004

[5] G. D. Abowd, M. Ebling, G. Hunt, H. Lei and H. W. Gellersen, "Context-Aware Computing", *IEEE Pervasive Computing*, pp 22 - 23, Volume 1, Issue 3, July-Sept. 2002

[6] J. Pearl, "Belief Networks Revisited". In *Artificial intelligence in perspective*, pp 49-56, 1994

[7] K. Henricksen, J. Indulska, A. Rakotonirainy, "Modeling Context Information in Pervasive Computing Systems", *First International Conference on Pervasive Computing, Pervasive'2002*, LNCS (2414), pp 167-180, August 2002.

[8] N. Friedman , L. Getoor , D. Koller, A. Pfeffer, "Learning Probabilistic Relational Models", *Proceedings of the 16th International Joint Conference on Artificial Intelligence*, pp 1300-1307, August 1999.

[9] D. Koller, A. Pfeffer, "Probabilistic frame-based systems", *Proceeding of the 15th National Conference on Artificial Intelligence*, pp 580-587, July 1998.

[10] W3C, "Web Ontology Language (OWL)", http://www.w3.org/2004/OWL/

[11] Jena, "A Semantic Web Framework for Java", http://jena.sourceforge.net/

[12] G. D. Abowd, A. K. Dey, "Towards a Better Understanding of Context and Context-Awareness", In *Proceeding of the Workshop on the What, Who, Where, When and How of context-awareness at CHI 2000*, April 2000.

[13] H.Lei, D. M. Sow, J. S. Davis, G. Banavar, M. R. Ebling, "The design and applications of a context service", *ACM SIGMOBILE Mobile Computing and Communications Review*, Volume 6, Number 4, pp 44-55, 2002.

[14] P. Gray, D. Salber, "Modeling and using sensed context in the design of interactive applications", In *Proceedings of 8th IFIP Conference on Engineering for Human-Computer Interaction*, Toronto, 2001.

[15] T. Gu, H. Pung, D. Zhang, "A Bayesian approach for dealing with uncertain contexts", *Proceedings of the Second International Conference on Pervasive Computing* , April 2004.

[16] The Microsoft Belief Network software (MSBNx), http://research.microsoft.com/adapt/MSBNx/