TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 13/20 pp707-713
Volume 12, Number 6, December 2007

Ontology-Based Context-Aware Middleware for Smart Spaces*

QIN Weijun (秦伟俊), SHI Yuanchun (史元春)**, SUO Yue (索 岳)

Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China

Abstract: Context-awareness enhances human-centric, intelligent behavior in a smart environment; however, context-awareness is not widely used due to the lack of effective infrastructure to support context-aware applications. This paper presents an agent-based middleware for providing context-aware services for smart spaces to afford effective support for context acquisition, representation, interpretation, and utilization to applications. The middleware uses a formal context model, which combines first order probabilistic logic (FOPL) and web ontology language (OWL) ontologies, to provide a common understanding of contextual information to facilitate context modeling and reasoning about imperfect and ambiguous contextual information and to enable context knowledge sharing and reuse. A context inference mechanism based on an extended Bayesian network approach is used to enable automated reactive and deductive reasoning. The middleware is used in a case study in a smart classroom, and performance evaluation result shows that the context reasoning algorithm is good for non-time-critical applications and that the complexity is highly sensitive to the size of the context dataset.

Key words: context-aware system; smart spaces; ontology; context model; first-order probabilistic logic

Introduction

Smart spaces are open, distributed, heterogeneous pervasive computing systems which aim to create a ubiquitous, human-centric environment with embedded sensors, information appliances, and multimodal interfaces that assist humans to efficiently complete tasks by offering abundant information and assistance. Context plays an important role in smart spaces in providing information about the status of the people, activities, location, physical environment, and computing entities. Applications in smart spaces use contextual information to become context-aware of changing situations relevant to the intelligent interactions with users[1,2].

For the last decade, research on context-aware computing domains has investigated a wide range of theoretical issues related to context modeling and representation, context inference mechanisms, and related development problems in practical systems such as data sensing, context acquisition, context management, and querying. However, the design and implementation of context-aware applications are still difficult. Contextual information requires a formal, unified model with rich expression power to represent context semantics and support an inference mechanism. Contextual information also requires infrastructure-level software supports to provide context acquisition, interpretation, management, and querying functions.

This paper presents an ontology-based approach for developing context-aware middleware for the following reasons[3,4]: (1) A well-defined, unified ontology enables knowledge sharing and reuse. (2) Ontologies with declarative semantics provide multiple policies to

Received: 2006-12-29; revised: 2007-06-21

∗ Supported by the Basic Research Foundation of Tsinghua National Laboratory for Information Science and Technology (TNList) and the National High-Tech Research and Development (863) Program of China (No. 2006AA01Z198)

∗∗ To whom correspondence should be addressed
E-mail: shiyc@tsinghua.edu.cn; Tel: 86-10-62772472

support context inference. (3) Ontologies provide various complex efficient inference mechanisms to deduce high-level contexts from low-level, raw context data, and to check inconsistent contextual information due to imperfect sensing. (4) Explicitly represented ontologies enhance the development of context-aware systems with semantic web technologies.

This paper introduces an ontology-based context model which describes contextual entities in a smart space environment and a context inference mechanism based on first-order probabilistic logic with an agent-based, loose coupling middleware that enables the acquisition, inference, management, and querying of contextual information from different sensors and software entities, to provide appropriate context-aware services for various applications.

1 Context Modeling and Reasoning

A well-defined context model is an important key to access the context in any context-aware system[5]. For instance, Henricksen[6] investigated the unified modeling language (UML) and entity-relationship (ER) modeling approach to represent context structures and properties. Gu et al.[7] presented an ontology-based context model to derive high-level contexts from low-level context data. The important context sources are captured from embedded sensors in a smart space environment which give uncertain, imperfect data. However, most of ontology-based context models fail to represent uncertainty, while logic-based context models fail to describe semantic relationships between context entities[5]. Here, the fundamental ontology-based and logic-based context models are combined in a first-order probabilistic logic to represent the basic context structure and construct a probabilistic inference mechanism, which combines the expressive power of first-order logic with the uncertainty context reasoning of probabilistic theory[8]. This shared understanding of specific domains gives context modeling which uses ontology and semantic web services to describe the concepts and relationships of context entities in smart spaces[9].

1.1 Formal representation of context

First order probabilistic logic (FOPL) is used to represent the basic context structure which combines first order logic and probabilistic models in a machine learning community. The definitions of terminology, including Field, Predicate, ContextAtom, and ContextLiteral, are presented in the following.

- $\text{Field} \in F^*$, where a Field is a set of individuals belong to the same class, e.g., Presson= {QIN, SHI, SUO}, Room={Room526, Room527}.
- $\text{Predicate} \in V^*$, where a Predicate indicates the relationship among the entities or the properties of an entity, e.g., location, co-locate.
- $\text{ContextAtom} \in A^*$, where ContextAtom is represented as predicate(term, term,...) in which a term is a constant, a variable, or a function followed by a parenthesized list of terms separated by commas with a predicate acting on the terms. For example, location(QIN) indicates QIN's location, and co-locate(QIN,SUO) indicates that QIN and SUO are located in the same place.
- $\text{ContextLiteral} \in L^*$, where ContextLiteral is represented as the form of contextAtom=v in which ContextAtom is the instance of ContextAtom and v indicates the status of ContextAtom or the value of the terms. For example, location(QIN)=Room527 indicates that QIN's location is Room527.

Thus, the context knowledge is represented as the form of $\Pr(L_1, L_2, L_3, ...) = c$, where $L_1, L_2, L_3 \in$ ContextLiteral, which indicates the concurrent probability of several ContextLiterals. For example, Pr(location (QIN)=Room527, location(SUO)=Room527) =0.76 indicates that the probability of the fact that QIN is located in Room527 while SUO is located in Room527 equals 0.76. The structures and properties of this basic model are described in an ontology language to define the conceptual contexts in a rich semantic level. The basic context structure is represented using web ontology language (OWL). Two OWL classes are defined as PriorProb and CondProb as in the approach of Ding et al.[10] for representing probabilities. A prior probability $\Pr(L_1)$ of a ContextLiteral L_1 is defined as the instance of class PriorProb, which has the two mandatory properties, hasContextLiteral and hasProbValue. A conditional probability $\Pr(L_1|L)$ of a context literal L_1 is defined as the instance of class CondProb, which has three mandatory properties hasContextLiteral, hasProbValue, and hasCondition. An example of an FOPL statement is represented by an OWL description is given in Fig. 1.

```
<owl:CondProb rdf:ID=" Pr(L1|L)">
    <hasCondition>L</hasCondition>
    <hasContextLiteral>L1</ hasContextLiteral >
    <hasProbValue>0.76</hasProbValue>
</owl:CondProb>
```

Fig. 1 FOPL statement represented in an OWL description

1.2 Ontology-based context model

The purpose of the ontology-based context model is to formalize the structured contextual entities in smart spaces by making use of the ontology methodology to define the concepts and relationships of the context elements. The context ontology is divided into a core context ontology for general conceptual entities in the smart space and an extended context ontology for the domain-specific environment, e.g., the classroom domain. The core context ontology defines very general concepts for the context in the smart space that are universal and sharable for building context-aware applications. The extended context ontology defines additional concepts and vocabularies for supporting various types of domain-specific applications.

The core context ontology describes seven basic concepts: user, location, time, activity, service, environment, and platform, which are the basic entities in the smart space as shown in Fig. 2. Part of the core context ontology is adopted from various widely-used consensus ontologies, such as DAML-Time and OWL-S. The instance of the smart space consists of User, Location, Time, Activity, Service, Environment, and Platform classes.

- User: Since user plays an important, central role in smart space applications, this ontology defines the vocabularies representing profile information, contact information, user preference, and user's mood which are sensitive to the current user activity or task.
- Location, Time, and Activity: The relationships between location, time, and activity facilitate validation of inconsistent contextual information when these contexts are sensed by sensors with different accuracies.
- Platform and Service: The platform ontology defines descriptions and vocabularies of hardware devices or sensors and software infrastructure in a smart space. The service ontology defines the multi-level specifications of services that the platform provides to support service discovery and composition. In the semantic web community, OWL-S is used as a standardized framework to describe services in general.
- Environment: The environment ontology defines the context specification of the physical environment conditions around the user, such as noise level, lighting condition, humidity, and temperature.

The extended context ontology extends the core context ontology and defines the details and additional vocabulary which applies to the various domains. The advantage of the extended context ontology is that the domain separation reduces the scale of the context knowledge and context processing burdens for pervasive computing applications, and facilitates effective context inference with limited complexity[9].

1.3 Probabilistic context reasoning

The context reasoning domain uses a rule-based inference mechanism which uses knowledge-based model construction (KBMC) to deduce high-level, new context knowledge from low-level detected facts. Within the FOPL framework, context rules are defined in the form of $\Pr(L_h \mid L_{b_1}, L_{b_2}, \ldots) = c$:-$L_{C_1}, L_{C_2}, L_{C_3}, \ldots$, which means that the probability of L_h is c for the constraints L_{C_1}, L_{C_2}, L_{C_3}, and conditions L_{b_1} and L_{b_2}. Note that L_{C_i} denotes only the context fact and the others denote arbitrary ContextLiterals. For instance, in the classroom scenario, the statement

Pr(TeacherStatus(Teacher)=
talking|Speaking(Student)=false)=
0.7:-IsBlackboardTouched(Room527)=false

denotes the rules that when the blackboard of Room527 has not been touched, the probability that the teacher is talking equals 0.7 for the condition that the student is silent.

After representing the context rule, an additional property element rdfs:dependOn is defined to capture the dependency relationship between the datatype and the object properties in OWL. The importance of the FOPL-based context rule is that probability and Bayesian network tools can then be used to reason with uncertain context. The benefit of the rdfs:dependOn element is used to translate the resource description

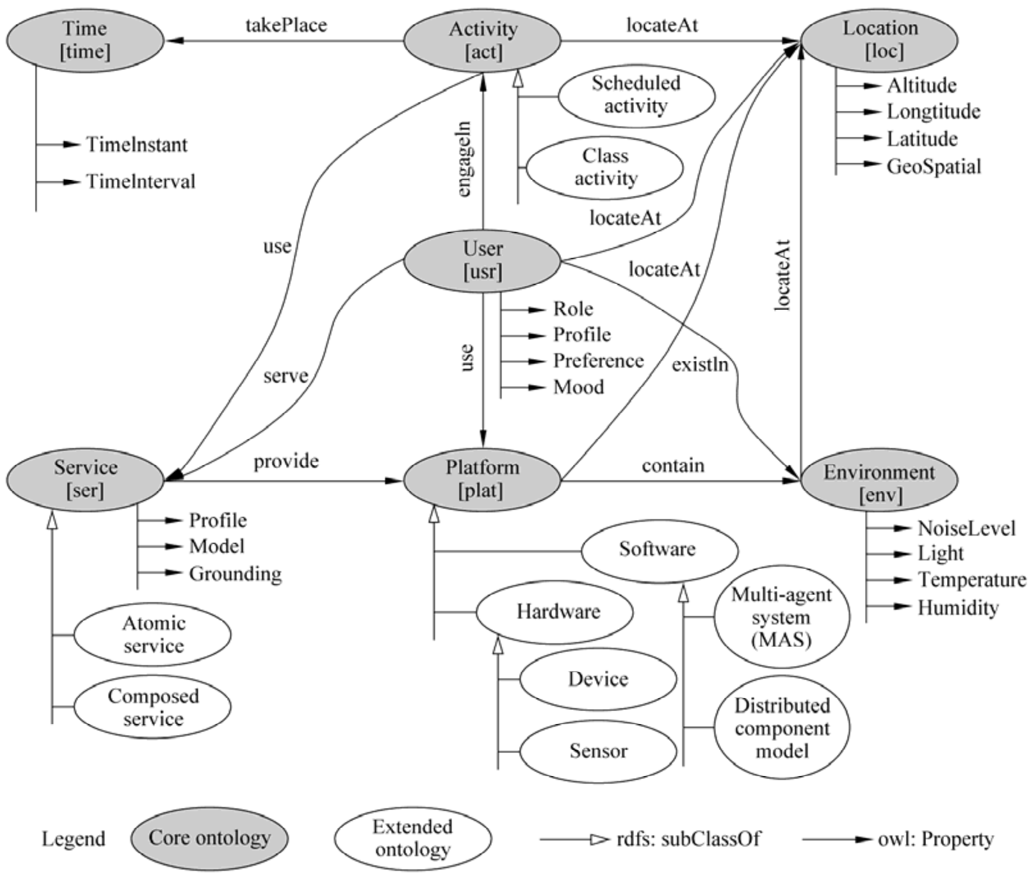

Fig. 2 Smart space context ontology

framework (RDF) graph into the Bayesian network's direct acyclic graph (DAG). Each node of the DAG represents a ContextLiteral, with directed arcs between nodes representing causal dependencies between ContextLiterals. The scale of the DAG is reduced by constraints, including valid syntax rules, the independence hypothesis of a causal set that extends the causal independence definition[11], the average distribution hypothesis of the residual probability, and the conditional independence hypothesis. These present the generation of the unnecessary nodes in the net so as to minimize the scale of the DAG and ascertain the exclusivity of the answer distribution. Therefore, the constraints and hypotheses control the inference complexity within an acceptable range.

2 System Architecture

2.1 Agent model

This ontology-based context-aware middleware is designed based on a multi-agent system which aims to support applications that make use of contextual information in a smart space environment. The agent model consists of several individual, collaborating agents as depicted in Fig. 3.

- Context wrapper agent: Acquires various types of raw context data from different sensors, devices, profiles, and software agents.
- Context provider agent: Abstracts context data from heterogeneous sources via various types of context wrapper agents and represents contextual information using ontologies for knowledge sharing and reuse.
- Inference engine agent: Provides an inference mechanism, including the reactive method, first order probabilistic logic and Bayesian networks, to infer high-level context from low-level data.
- Knowledge base agent: Stores inference rules, observed facts, and ontologies for context data management and maintenance in the database.
- Query filter agent: Provides query interfaces for upper applications or agents to query or subscribe the

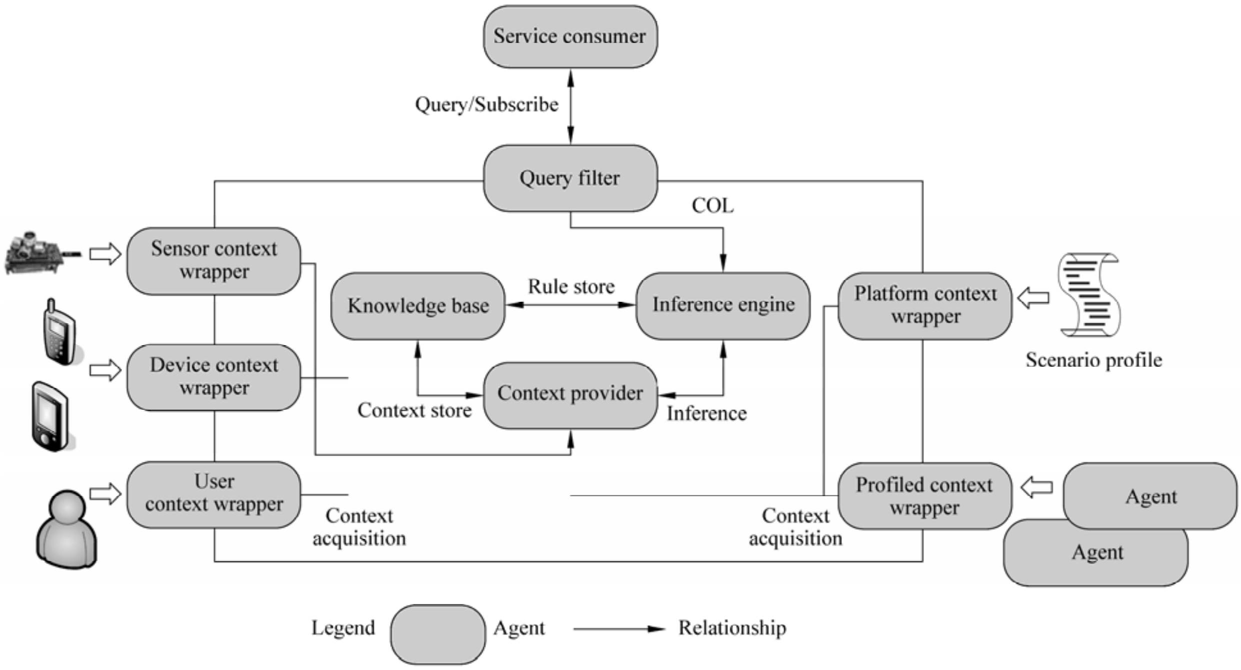

Fig. 3 Middleware architecture

context-aware services with support of a system-level coordination mechanism using a formal query language.

2.2 Design considerations

The most important considerations in the architecture design for providing context-aware services are:

- **Loose coupling** Contextual information in a smart space environment is very dynamic and heterogeneous. With this loose coupling, the system can use suitable plug-in components to meet the different demands of context-aware applications for modeling and reasoning with different types of context knowledge with the least system integration costs.
- **Scalability** The middleware architecture with component abstraction and encapsulation provides an easy way to enable context-aware service scalablility. By customizing the scenario profile and deploying various types of sensors, the context wrapper agent can capture abundant contextual information from different sources to be more adaptive to the real smart space environment.
- **Invisibility** By separating the application procedure and the underlying services, the middleware can use a query filter agent module for enabling underlying system functionalities (e.g., context data storage, sensor distribution, inference engine) that is invisible to the upper applications.

3 Validation

The preceding section described the design of the ontology-based context-aware middleware. The middleware was validated by applying it to a case study in the smart classroom project[12,13] to demonstrate its usability. The context reasoning performance was also evaluated.

3.1 Case study

For the smart classroom project, a smart cameraman module was designed to change the live-video scene to a situational context according to the class activity in a classroom by switching an array of cameras. By making use of context-aware services, remote students were able to focus their attention on relevant scenes on the client side. In this case, the context-awareness provided by the middleware captures the contextual information relevant to the user's activity and provides class activity clues to the smart cameraman module. The middleware also delivers customized video to remote students with various quality due to the various capabilities of their computer or systems, such as display screen size or network bandwidth.

The smart cameraman scenario uses four types of context rules in the class activity:

- **Teacher writing on the MediaBoard** When the teacher is writing comments on the MediaBoard, the smart cameraman module may select a close-up view of the board, as shown in Fig. 4a.
- **Teacher showing a model** When the teacher holds up a model, the smart cameraman module may zoom in on the model as shown in Fig. 4b.
- **Remote student speaking** When a remote student is speaking, live video of the student may be delivered to other remote students.
- **Other** In all other situations, the smart cameraman module may select an overview of the classroom shown in Fig. 4c.

A predefined probability between 0 and 1 is attached to the context rules using basic structure of the first-order probabilistic logic partially shown in Table 1. When an event (e.g., the teacher writing on the mediaboard) occurs, the concurrent probability distribution of the camera's status is reconstructed according to the context rules, so that the camera tracks the live focus. A case generator shown in Fig. 4d has been developed to simulate a variety of situations and contextual information to test the functionalities of the smart cameraman module.

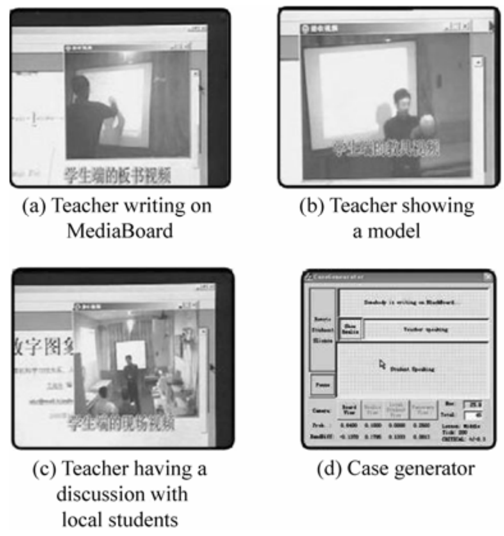

Fig. 4 Case study in the smart classroom project

Table 1 Examples of context rules defined in the smart cameraman scenario

Context rules	FOPL formula
CR1	Pr(IsStatus(Camera)=CLOSEUP_VIEW|Action(Tearcher)=WRITING, IsStatus(MediaBoard)=TRUE)=0.8:OnTouched(MediaBoard)=TRUE
CR2	Pr(IsStatus(Camera)=CLOSEUP_VIEW|Action(Tearcher)=SHOWING)=0.15:OnTouched(MediaBoard)=TRUE
CR3	Pr(IsStatus(Camera)=CLOSEUP_VIEW|Action(Tearcher)=SPEAKING)=0.05:OnTouched(MediaBoard)=TRUE

3.2 Performance evaluation

This section illustrates the performance evaluation of the context reasoning algorithm. The goal of the tests is to evaluate the computational complexity and run-time performance of the context reasoning algorithm. The middleware uses the JENA2 Semantic Web Toolkit to build the context reasoner which supports the rule-based inference in OWL. The context entity datasets were small 1000 RDF triples to large 5000 RDF triples for reasoning with different computational capabilities (1 GB RAM with P4/2.0 GHz and P4/3.2 GHz) using context rules described in the previous section.

Figure 5 shows the results including the size of the Bayesian network using the context data. Figure 6 shows the runtime quantitative delay of the context reasoning. The results show that the context reasoning algorithm is good for non-time-critical applications, and that the complexity is highly sensitive to the size of the context dataset. With the dataset using 5000 RDF triples, the context reasoning results were delayed by over 2 s on a P4/3.2-GHz CPU. Therefore, for time-critical applications such as the smart cameraman module, the scale of the context dataset and the context rules should be controlled.

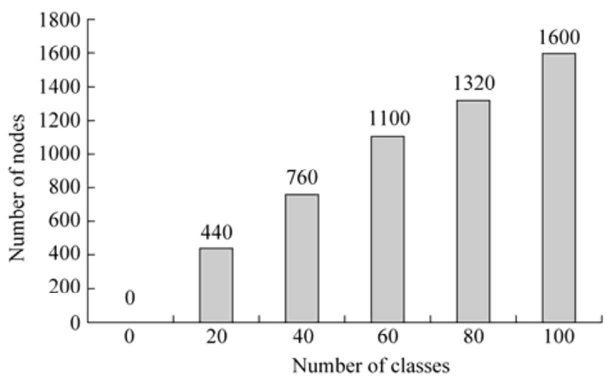

Fig. 5 Size of the Bayesian network

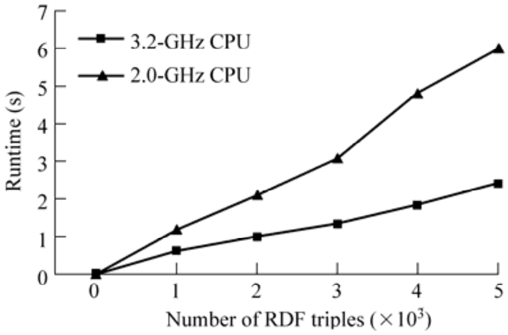

Fig. 6 Context reasoning runtime performance

4 Conclusions

A context-aware middleware was developed to provide context-aware services for smart spaces. The middleware supports the high-level abstraction of contextual information with the power of a formal context model which combines first-order probabilistic logic and explicitly represented ontologies and allows context inference based on an extended Bayesian network to provide more precise context information adapted to changing, heterogeneous smart space environments. Further research will investigate description logic approaches with more expressive power to make the middleware more robust and extensible.

References

[1] Dey A K, Salber D, Abowd G D. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human Computer Interaction* (*HCI*) *Journal*: *Special Issue on Context-Aware Computing*, 2001, **4**(2-4): 97-166.

[2] Dey A K. Providing architectural support for building context-aware applications [Ph. D. Dissertation]. Georgia Institute of Technology, Georgia, USA, 2000.

[3] Chen H, Finin T, Joshi A. An ontology for context-aware pervasive computing environments. *Knowledge Engineering Review*: *Special Issue on Ontologies for Distributed Systems*, 2004, **18**(3): 197-207.

[4] Chen H. An intelligent broker architecture for pervasive context-aware systems [Ph. D. Dissertation]. University of Maryland, Baltimore, USA, 2004.

[5] Strang T, Linnhoff-Popien C. A context modeling survey. In: Proceedings of UbiComp Workshop on Advanced Context Modeling, Reasoning and Management. Nottingham, UK, 2004.

[6] Henricksen K. A framework for context-aware pervasive computing applications [Ph. D. Dissertation]. University of Queensland, Brisbane, Australia, 2003.

[7] Gu T, Pung H K, Zhang D Q. A service-oriented middleware for building context-aware services. *Journal of Network and Computer Applications*, 2005, **28**(1): 1-18.

[8] Poole D. First-order probabilistic inference. In: Proceedings of IJCAI. Acapulco, Mexico, 2003: 985-991

[9] Guarino N. Formal ontology and information systems. In: Proceedings of FOIS. Trento, Italy, 1998: 3-15.

[10] Ding Z L, Peng Y, Pan R. Soft Computing in Ontologies and Semantic Web. Berlin: Springer-Verlag, 2005.

[11] Rish I, Dechter R. On the impact of causal independence. In: Proceedings of Stanford Spring Symposium on Interactive and Mixed-Initiative Decision Theoretic Systems. California, USA, 1998: 101-108.

[12] Shi Y C, Xie W K, Xu G Y, Shi Y T, Chen E Y, Mao Y H, Liu F. The smart classroom: Merging technologies for seamless tele-education. *IEEE Pervasive Computing Magazine*, 2003, **2**(2): 47-55.

[13] Xie W K, Shi Y C, Xu G Y, Mao Y H. Smart platform—A software infrastructure for smart space (SISS). In: Proceedings of International Conference on Multimodal Interfaces. Pittsburgh, USA, 2002: 429-436.