

# From the OntoBayes Model to a Service Oriented Decision Support System

Yi Yang and Jacques Calmet  
Institute for Algorithms and Cognitive Systems (IAKS)  
University of Karlsruhe (TH)  
76131 Karlsruhe, Germany  
{yiyang,calmet}@ira.uka.de

## Abstract

*The aim of this paper is to propose a service oriented decision support system based on an ontology-driven uncertainty model (OntoBayes). OntoBayes consists of knowledge and decision model parts. The former is the integration of ontologies and Bayesian Networks while the latter can describe different decision models. OntoBayes gives a solution to deal with uncertainty and structure complexity and provides decision models by decision making. In order to construct a decision support system, a service oriented framework and architecture will be introduced finally.*

## 1. Introduction

During the 1950s and 1960s the concept of decision support was investigated from two aspects: the theory of organizational decision making and the techniques of interactive computer systems. After that it became an area of research that focuses on computerized system supporting decision making activities [1]. A decision support system (DSS) can be defined as “a computer program that provides information in a given domain of application by means of analytical decision models ...” [2]. Nowadays there are different types of DSS at the conceptual level: communication-driven DSS, data-driven DSS, document-driven DSS, knowledge-driven DSS, and model-driven DSS [3].

In this paper, we propose a service oriented DSS. First, we investigate an ontology driven uncertain model, OntoBayes, consisting of knowledge and decision model parts. The former is the integration of ontology and Bayesian Networks (BN) while the latter can describe different decision models. After that a service oriented design of DSS at a high level is introduced. The selected overall framework for this DSS is the multiagent paradigm.

The remaining sections of this paper are structured as follows. Section 2 scratches the surface of the agent approach for web services and decision making. Section 3

introduces an ontology driven uncertain model OntoBayes. This section differs partly from previously published results since it is now fully suited to decision support. Section 4 is devoted to investigate the design of a service oriented decision support system based on OntoBayes. Section 5 surveys related works and finally, Section 6 contains a brief overview on future work and concludes the paper.

## 2. Agent, Web Services and Decision Making

In the cyber world, agent based approaches are more powerful when running agents in a distributed and dynamic environment (potentially on a web-wide scale) to perform complex actions on behalf of their users [4]. Unifying agents and web services can enhance the construction and flexibility of web service applications [5].

Before introducing other agent based approaches in the following sections, it is necessary to give some agent related definitions hereafter. An agent is capable of autonomous action in situated environment in order to meet its design objectives [6]. Based on this definition an intelligent agent can be extended with three additional characteristics: reactivity, proactivity and social ability. The concept of multiagent has emerged as a paradigm for designing complex software systems. It is used to better formalize problems in Distributed Artificial Intelligence (DAI) [7].

The world of web services was characterized as loosely-coupled distributed systems based on service oriented computing (SOC). The use of web services could be considered as actions that the agent may take to meet its goals. In [8] four major trends in internet computing were analyzed which are able to driven SOC and Multi Agent Systems (MAS) research in the future. There are some emerging approaches with MAS-like characteristics in SOC, such as ubiquitous computing, ontologies, service-level agreements and quality-of-service measures etc. All of them can be perfectly performed with MAS concepts and techniques.

In general, decision making is not a simple event but a process leading to the selection of a course of action among

several alternatives. Agents need to select and to compose different actions in order to make a decision. From the point of view of service, the process of decision making consists of different services provided by the agent itself or by external agents. This viewpoint requires agents to unify services and agents themselves. Agent should be able to publish, discover, select and compose services in the cyber world, in order to meet their goals.

### 3. The OntoBayes Model

The normal ontological model is not sufficient to express the domain specific uncertainty or risks. In order to overcome this shortcoming the OntoBayes model, an ontology-driven uncertainty model, was proposed [9]. This model integrates the Bayesian approach into ontologies and to preserve the advantages of both. It consists of knowledge and decision model parts.

#### 3.1. Basic Concepts in OntoBayes

The basic concepts or methodologies used to construct the OntoBayes model are ontologies, Bayesian Networks, influence diagrams and multi-attribute utility theory.

Ontologies can formally and explicitly specify a shared conceptualization [10]. OntoBayes makes use of OWL<sup>1</sup> as its formal language to facilitate its knowledge representation. Ontologies can excellently represent the organizational structure of large complex domains, but their application is bounded because of their inability to deal with uncertainty [11]. In order to overcome this limitation, Bayesian Networks are used in this model. Bayesian Networks are widely used graphical model for probabilistic knowledge representation under uncertainty [12]. Two kinds of information can be represented in BN: structure and numerical information. The former is in principle a directed acyclic graph (DAG) where nodes are the random variables and arcs between nodes imply the dependency (or conditional) relation between the variables. The latter is the Bayesian probabilistic information of random variables.

With the integration of ontologies and Bayesian Networks the OntoBayes model can build its knowledge part. To complete the task of decision making support another part, the decision model part, is required. This part contains all decision models that can be defined through different methods. One of them is influence diagrams (ID). An influence diagram is a graphical representation based on Bayesian networks, but with different types of nodes and arcs. While BN has only one type of node (chance node) and arc (conditional arc), ID has three types of nodes: chance node, decision node and utility node, and two kinds

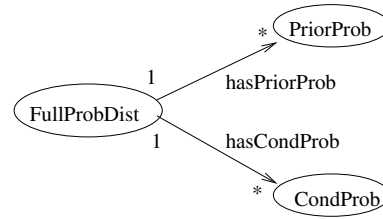


Figure 1. The upper ontology for the probabilistic extension.

of arcs: conditional and informational arcs. The decision node is used to specify the choices available to the decision maker and the utility node associated utility function represents the objective to be maximized in expectations. Arcs in decision nodes are informational. They only imply time precedence and have no conditional meaning [13].

In order to evaluate utility nodes in ID OntoBayes makes use of multi-attribute utility theory (MAUT), because a utility node may conditionally depend on more than one attributes. Generally utility theory is a systematic approach to quantify the preferences of an individual. It rescales a numerical value on some measures of alternatives onto a 0-1 scale with 0 representing the worst preference and 1 the best one. A rank as an end result will be evaluated to reflect preferences. MAUT is a decision analysis tools designed to handle the tradeoffs among multiple objectives [14].

#### 3.2. OWL annotations for Bayesian Probability

To express uncertain information or the risk of actions we need an extension of OWL which enables to specify probability-annotated classes or properties. The extension can be simply specified with an upper ontology in Figure 1.

We define three OWL classes: “PriorProb”, “CondProb” and “FullProbDist”. The first two classes are used to define prior probability and conditional probability respectively. They have a same datatype property “ProbValue”, which can express the probabilistic value between 0 and 1. The last one is used to specify the full disjoint probability distribution. It has two disjoint object properties: “hasPrior” or “hasCond”.

The property “hasPrior” specifies the relationship between classes “FullProbDist” and “PriorProb”. It indicates that the instances of “PriorProb” are elements of one instance of “FullProbDist”. The property “hasCond” describes the relationship between the classes “FullProbDist” and “CondProb” and it has a similar semantic meaning as “hasPrior”. These two properties are disjoint since any instance of “FullProbDist” can have only one probability

<sup>1</sup><http://www.w3.org/2004/OWL/>

type: either prior or conditional.

### 3.3. OWL annotations for dependency

The probabilistic extension of OWL alone is not enough for modeling our ontology-driven BN. It is required to specify the dependency relations between the random variables explicitly.

To solve this problem an additional property element `<rdfs:dependsOn>` was introduced in order to markup dependency information in an OWL ontology. Before the relation of dependency is formally defined, it is necessary to introduce some notations which are influenced by the Object Oriented Programming (OOP) approach. In OWL there are two type of properties: object property and datatype property. We denote an object property  $s$  between domain class  $A$  and range class  $B$  as  $s(A, B)$ . It is considered as an available operation of the subject class  $X$  to object class  $Y$ . A datatype property  $d$  of class  $A$  will be denoted as  $A.d$ , where  $d$  is considered as an attribute of  $A$ . Now the dependency between properties can be defined as follows.

**Definition 3.1** A dependency is a pair  $X \rightarrow Y$ , where each of  $X$  and  $Y$  is either a datatype property  $X.d$  or an object property  $s(X, Y)$ . It is read as “ $X$  depends on  $Y$ ”.

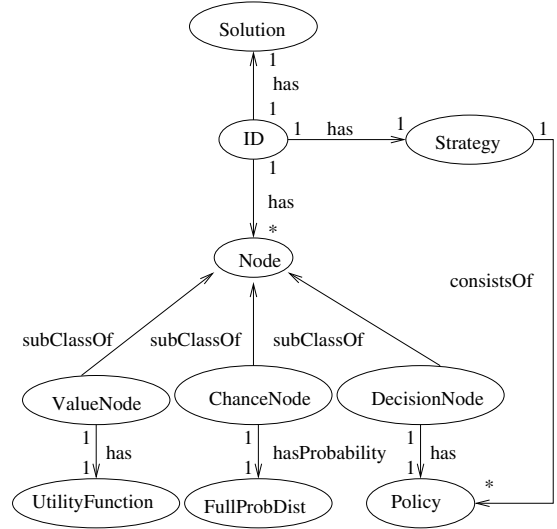
This definition clearly points out that each random variable in OntoBayes modeled BN is either an object property or a datatype property. The dependency relation was not specified between classes but between properties, because this avoids possible errors when extracting a Bayesian structure from ontologies [9].

### 3.4. OWL Annotations for Influence Diagrams

Decision models will be comprehensible for agents, only when they have a semantic understanding of their structure. Upper ontologies can satisfy this requirement of facilitating agents to recognize different decision models. In Figure 2 an upper ontology is depicted to define influence diagrams.

From the upper ontology, a domain specific ontology for ID describes the vocabularies used in this domain. Here, it is clearly pointed out that an ID has a solution, strategy and three type nodes. The value (or utility) node is associated to a utility function and the chance node reflects a probabilistic distribution (prior or conditional). The decision node is associated with a policy which is used to generate strategies. The cardinality specifies the numerical constraints on each ontological concept. By using this domain specific template, an ID can be generated automatically with OWL.

The annotation for ID in OWL can be similarly specified as the probabilistic extension mentioned in Section 3.2.



**Figure 2.** The upper ontology for influence diagrams.

**Table 1.** The four layer framework for DSS

Application Layer
Service Layer
Management Layer
Repository Layer

## 4. Service Oriented DSS

In this section, a decision support system is presented. The design of the system consists of two aspects: framework and architecture. Both of them are service oriented.

### 4.1. The framework for DSS

The general framework for the system is made of four layers that are showed in Table 1. Each layer has its own functionalities and can communicate with the layers that are directly adjacent to it. In a bottom up approach the hierarchical layers construct a system from the back end to the application front end.

The lowest layer, the repository layer, consists of different kinds of repositories. A repository is a place to store and maintain data; it can be centralized or distributed over a network. Traditionally, the lowest layer in a system is the database layer which has a similar functionality. Nowadays the trend is to build repositories within the system rather than to use simple databases because more and more information systems are not simply data-based, but heterogeneous and hybrid. They cover different types of information

resources such as knowledge, models and services etc. It is impossible to store them in a unique database, but it is easy to classify, store, maintain and access them through different kinds of repositories.

The management layer is the most important and complicated layer of the framework. It is the backbone of the system and sustains the running system. There are a lots of components required to provide different operations and functionalities to the system. From the point of view of a service oriented system these components must be loosely coupled and independently managed, in order to insure a flexible configurability, lasting lifetime and granularity of the system. Each management provides its own services that can be used within the system or be published to construct a business process for instance. In this layer one finds at least the repository management which is responsible for accessing, storing and maintaining different repositories.

The service layer connects the back end and the application front end of the system. In this layer the services provided by the system are described and published for the application front end. Applications such as business processes or decision making processes can select different services on demand and compose them into a whole process. This layer insures high quality performances of the system according to the features of a service oriented architectures (SOA): loose coupling, implementation neutrality, flexible configurability, long lifetime and granularity [15].

The application layer is the front end of the system. Users or agents will discover and select services provided by the system to build their own applications. Applications can be used either within the constructed new system to design different management tools or for third parties which are the service consumers located outside the system, e.g. a knowledge worker from a partner company who want to access the knowledge repository.

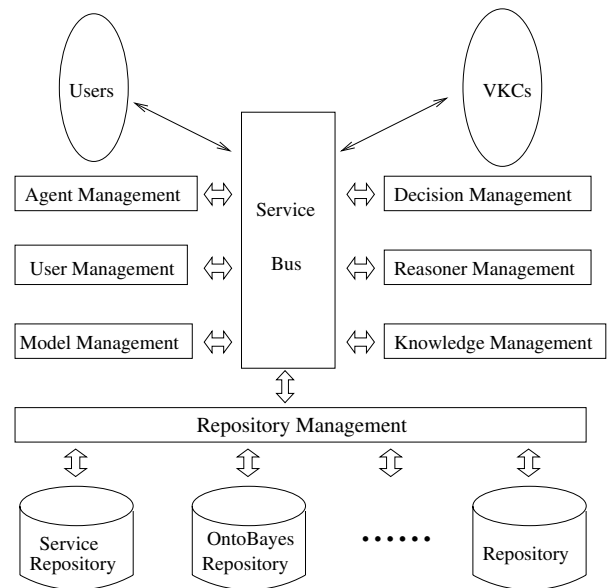
## 4.2. The architecture for DSS

According to the general framework described above it is now possible to design a concrete architecture for the decision support system. In Figure 3 a service oriented architecture for the system is depicted. The whole system consists of different components and each of them possesses its own layers underlying the four layer framework.

In the repository layer there are different kinds of repositories. Among them the service and OntoBayes Repository are the most important:

- *Service repository* is used to store service descriptions. Here an XML format language, the web services description language (WSDL)<sup>2</sup>, will be used to describe different services.

<sup>2</sup><http://www.w3.org/TR/wsd1>



**Figure 3. An architecture for the decision support system.**

- *OntoBayes repository* is the storage for the OntoBayes model which includes the knowledge base and decision models. Agent can get all kinds of ontology based information from this repository with or without corresponding upper ontologies.
- Other repositories can also be built for the system on demand. For example repositories for storing policy, business processes and users (or agents) information.

In the management layer there are several types of management components and each of them has its own functionalities and provides different services.

- *Repository management* can provide basic services to manage all kinds of repositories, e.g. access, recovery and long term preservation of repositories. All additional functionalities or services to different repositories will be provided from other management components.
- *Decision management* takes responsibility for managing the activities related to decision making. For example management of different decision making processes, to distinguish different decision situations (single decision vs. social decision making), evaluation of decisions, and so on.
- *Model management* is used to control different decision models. It provides all services related to decision models such as commit, edit, delete etc ...

- *Knowledge management* can provide services to structure, exchange, integrate and maintain knowledge. For example, the new knowledge piece submitted from an agent will be controlled here first, and then can be committed to the repository.
- *Reasoner management* provides different reasoning mechanisms. For example an agent can choose to use exact reasoning or approximate reasoning to solve a Bayesian structured problem.
- *User management* provides services for user access control. Each user has a different role in the system and has also different rights. For example, there are knowledge workers, knowledge experts, administrators, decision makers, end users, and so on. Users may consume services provided by the system only when they are granted permission.
- *Agent management* takes responsibility to manage all of the agents. There are two kinds of agents running on the system: management agent and user agent. The former runs on each management component and has its functionalities, the latter is assigned tasks by the user or may represent him in some situations.

In the service layer there is a service bus which enables a logical integration of service consumers and providers [16] and a logical connection between the back end and the application front end of the system.

The application layer consists of different applications which can be divided into two classes: The internal application and external application. An internal application refers to applications that belong to the system. For example the knowledge expert will edit the knowledge with a knowledge editor such as Protégé<sup>3</sup>. For different management components in the system there are different application tools. An external application refers to applications that can be used out of the system. For example virtual knowledge communities (VKCs) [17] provide the application platform to facilitate knowledge sharing between agents from inside and outside. They can support agents by decision making with corporate knowledge [18]. The external applications make use of services provided by different systems and combine them in order to satisfy their own requirements.

### 4.3. Features of the system

Based on the service oriented design the system can exhibit a lot of features:

- **Flexibility.** The DSS is configured flexibly because different components can be combined late in the decision making process. The change of configuration is dynamical.

<sup>3</sup><http://protege.stanford.edu/>

- **Reusability.** The system components are loosely coupled, so that they can be more easily used and combined based on decision makers' needs.
- **Scalability.** The system can be easily changed with additional services to satisfy more requirements or with reduced services to support small applications.
- **Implementation neutrality.** The performance of the system does not depend on the implementation details of the interacting components, but on the provided services.

There are other features that the system can possess. They are inherent to the standard architecture SOA. An overview of all these features is given in [15].

## 5. Related works

Numerous approaches to DSS have already been investigated but, to our knowledge, it does not exist an approach that improves DSS through the combination of uncertainty, agent and web services as we do.

Hendler proposed an idea for uniting agents and the semantic web in [4], that is also discussed in [5]. Gibbins et al. investigated an agent based semantic web service for situational awareness and information triage in a simulated humanitarian aid scenario [19].

There are some attempts to define probabilistic methodologies to represent uncertainty inclusion ontologies. Among them the approaches in [11] and [20] seem closest to ours. The first integrates BN into a frame-based system to preserve the advantages of both, but with totally different languages and methods. The second extends OWL using Multi-Entity Bayesian Networks (MEBN) to address the inability to represent, and reason on, uncertainties in the semantic web. This extension has more logical expressive power, but is limited for entity-based modeling of uncertain knowledge representation. In comparison, our approach allows uncertain knowledge modeling not at the conceptual level but at the property level, which enables to express actions formally and naturally in the form of "action:=subject-verb-object". Furthermore, none of both works does propose a formal notation to show how the integration syntactically and semantically works.

The work of modeling corporate knowledge based on agent oriented abstraction (AOA) was a major source of inspiration of this research. The AOA paradigm covers the concepts of agents, annotated knowledge, utility functions and society of agents. AOA assumes that agents are entities consisting of both a knowledge component and a decision making mechanism [21]. OntoBayes is thus designed in agreement with the AOA paradigm. In [22] the application of AOA model to the abstract modeling of corporate

knowledge is investigated. Corporate knowledge was defined as the amount of knowledge provided by individual agents. The concrete implementation for corporate knowledge within AOA was introduced as VKCs [17, 23].

## 6. Conclusion and future work

This paper proposed a service oriented approach in the area of research for DSS. A four layer framework and a concrete architecture were introduced. The core of the system is the OntoBayes model which enables to deal with uncertainty and decision analysis. This model is ontology driven, so that there are several extensions of OWL to satisfy different requirements. The influence diagram as a typical decision analysis method was introduced as the decision model part of OntoBayes. One must emphasize that this part can contain different decision models such as Markov decision process (MDP), decision network and so on. With domain specific upper ontologies any model can be described in OWL enabling thus a semantical understanding.

Although the OntoBayes model and the system design shown in this paper are important steps towards a service oriented decision support system, further investigations are still required. A system is being implemented for applications in insurances and natural disaster management. Some of the future research directions deal with:

- A trust mechanism for agents involved in uncertain knowledge sharing.
- Methods to assess achieved decisions based upon corporate knowledge with possible conflicting information.

## References

- [1] D. J. Power. A brief history of decision support systems. *DSSResources.COM*, 2003.
- [2] M. R. Klein and L. B. Methlie. *Knowledge-based Decision Support Systems*. Wiley, 1995.
- [3] D. J. Power. *Decision support systems: concepts and resources for managers*. Quorum Books, April 2002.
- [4] J. Hendler. Agents and the semantic web. *IEEE Intelligent Systems*, 21(2):30–37, 2001.
- [5] C. D. Walton. Uniting agents and web services. *AgentLink News*, (18):26–28, August 2005.
- [6] M. Wooldridge. Intelligent agents. In W. Gerhard, editor, *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, chapter 1, pages 27–78. The MIT Press, 1999.
- [7] G. Weiss. *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. MIT Press, 1999.
- [8] M. N. Huhns, M. P. Singh, M. Burstein, K. Decker, E. Durfee, T. Finin, L. Gasser, H. Goradia, N. Jennings, K. Lakkaraju, H. Nakashima, V. Parunak, J. S. Rosenschein, A. Ruvinsky, G. S. S. Swarup, K. Sycara, M. Tambe, T. Wagner, and L. Zavala. Research directions for service-oriented multiagent systems. *Internet Computing*, 9(6):65–70, November/December 2005.
- [9] Y. Yang and J. Calmet. Ontobayes: An ontology-driven uncertainty model. In *Proceeding of the International Conference on Intelligent Agents, Web Technologies and Internet Commerce (IAWTIC'05)*, volume I, pages 457–464, Vienna, Austria, November 2005. IEEE Computer Society.
- [10] T. R. Gruber. A translation approach to protable ontology specification. *Knowledge Acquisition*, 5:199–220, 1993.
- [11] D. Koller and A. Pfeffer. Probabilistic frame-based systems. In *Proc. AAAI National Conference on Artificial Intelligence (AAAI'98)*, pages 580–587, Madison, Wisconsin, USA, July, 1998.
- [12] F. V. Jensen. *An introduction to Bayesian networks*. UCL Press, 1996.
- [13] R. D. Shachter. Evaluating influence diagrams. *Operations Research*, 34(6):871–882, 1986.
- [14] J. Butler, D. J. Morrice, and P. W. Mullarkey. A multiple attribute utility theory approach to ranking and selection. *Management Science*, 47(6):800–816, June 2001.
- [15] M. P. Singh and M. N. Huhns. *Service-Oriented Computing*. Wiley, 2005.
- [16] M. Endrei, J. Ang, A. Arsanjani, S. Chua, P. Comte, P. Krogdahl, M. Luo, and T. Newling. *Patterns: Services-Oriented Architecture and Web Services*. IBM Redbooks, April 2004.
- [17] M. Hammond. Virtual knowledge communities for distributed knowledge management: A multi-agent-based approach using jade. Master's thesis, University of Karlsruhe and Imperial College London, 2004.
- [18] Y. Yang and J. Calmet. Ontobayes approach to corporate knowledge. In *Proceeding of the 16th International Symposium on Methodologies for Intelligent Systems (ISMIS'06)*, LNCS/LNAI. Springer, 2006.
- [19] N. Gibbins, S. Harris, and N. Shadbolt. Agent-based semantic web services. In *In Proceedings of Twelfth International World Wide Web Conference (WWW'03)*, Budapest, HUNGARY, May 2003. W3C, Association for Computing Machinery.
- [20] P. C. G. da Costa, K. B. Laskey, and K. J. Laskey. Pr-owl: A bayesian ontology language for the semantic web. In *Proceedings of the first workshop on Uncertainty Reasoning for the Semantic Web (URSW'05)*, pages 23–33, Galway, Ireland, November 2005.
- [21] J. Calmet, P. Maret, and R. Endsuleit. Agent-oriented abstraction. *Revista (Real Academia de Ciencias, Serie A de Matematicas)*, 98(1):77–84, 2004. Special Issue on Symbolic Computation and Artificial Intelligence.
- [22] P. Maret and J. Calmet. Modeling corporate knowledge within the agent oriented abstraction. In *Proc. International Conference on Cyberworlds (CW'04)*, pages 224–231. IEEE Computer Society, 2004.
- [23] P. Maret and J. Calmet. Corporate knowledge in cyberworlds. *IEICE Transaction, Information and Systems Society*, E88-D(5):880–887, May 2005.