# Semantic Processing in IT Management

Andreas Textor, Fabian Meyer, Reinhold Kroeger
*Distributed Systems Lab*
*RheinMain University of Applied Sciences*
*Unter den Eichen 5, D-65195 Wiesbaden, Germany*
{*firstname.lastname*}*@hs-rm.de*

***Abstract*—In the domain of IT management, numerous models, protocols and tools have been developed. To achieve the long-term goal of comprehensive, highly automated IT management, the various sources of information need to be combined. As syntactic translation is often not sufficient, ontologies can be used to unambiguously and comprehensively model IT environments including management rules. In this paper, we present an approach that combines the domain model, rules, instance data (which represents real-world systems) into an ontology. Moreover, probabilistic knowledge of the domain is modeled using Bayesian networks and integrated into the ontology. A runtime system that aggregates data and merges it into the ontology, and then uses a reasoner to evaluate management rules, is described as part of the approach of the ongoing project.**

***Keywords*-ontology; IT management; Bayesian network**

## I. Introduction

Knowledge bases grow in size and complexity in every domain. For this reason, in the domain of IT management, numerous models, protocols and tools have been developed. Notable models include the OSI network management model (also known as CMIP, the name of its protocol) and the still widely used simple network management protocol (SNMP). A more recent approach to specify a comprehensive IT management model is the Common Information Model (CIM, [1]), a widely recognized Distributed Management Task Force (DMTF) standard. The more complex an IT environment gets, the more important the capability becomes to automate as many tasks as possible. Both commercial and free management tools and frameworks exist that cover different parts of the required feature set for management tasks, but usually not only a single tool, but a set of tools is used. In order to achieve a unified view of the heterogenous integrated management models, mappings between different types of models can be defined. However, syntactic translations are often not sufficient, when the same concept is represented in a different way in different domains. This problem can be approached by using ontologies to clearly define the semantics.

Only when a comprehensive formal representation of the domain data exists, that is also capable of modeling rules, a largely automatic management becomes possible, because then not only structural, but also behavioural information is expressed in the model. To achieve such an automated management system, more prerequisites must be provided: A runtime system is required to import the corresponding domain model into the ontology and to evaluate the rules, based on up to date data from the managed system. Therefore, instance data must be acquired at runtime and added to the ontology, so that rules can be evaluated according to both model and instance data.

In certain cases, and especially in a domain as complex as IT management, the domain cannot be modeled solely using exact information, which might not be available. However, when relationships between entities are known and marked accordingly in the model, probabilistic evaluation is possible, where only incomplete data is available. To enable that, the ontology and the runtime system need to be extended accordingly.

The approach presented in this paper uses an OWL (Web Ontology Language, [2]) ontology to combine the domain model, instance data and rules defined in SWRL (Semantic Web Rule Language). To model entities and relationships of an IT environment, the CIM model was converted into an OWL ontology (the translation process is described in more detail in [3]). To model probabilistic knowledge, ontology elements are annotated so that a Bayesian network can be partially derived at runtime. Bayesian networks are a probabilistic model to specify causal dependencies between random variables in a directed acyclic graph.

Section II describes related work in the context of ontologies and IT management, and section III gives an overview of our approach. The paper closes with a conclusion in section IV.

## II. Related Work

There are several publications that examine the application of ontologies to the domain of IT management, e.g. [4], [5]. In [6] the authors provide mappings for parts of different IT management models to OWL, including Structure of Management Information (SMI) and the Common Information Model (CIM). The resulting ontology can be used to combine the knowledge given in the different representations into a joint model. One problem the authors point out for the mapping is information that can be expressed in the original languages, but has no direct representation in OWL, such as the attachment of measurement units or access authorizations

 ISBN: 978-1-61208-175-5 

to properties. To solve this problem, the data is presented on the Resource Description Framework (RDF) layer of OWL. In RDF, it is possible to attach additional information to edges in the graph so that the data can be represented.

[4] describes how to represent several abstraction layers of a system in split ontologies to achieve a pyramid-like structure of ontologies, where often used ontologies are at the bottom of the figure. The re-use of components and models is always an important topic in IT systems. The paper shows that OWL is capable of organizing several abstractions of a system in ontologies and reuse defined components in higher layers.

A real-world management application is shown in [5] where ontologies are used to manage a network infrastructure. SWRL rules are used to create new object property connections between entities in case of a blackout. For this, properties and instance structures are observed. As basis for the paper Policy-based Network Management (PBNM) [7] was used. Rules are evaluated periodically during runtime, and new facts are added to the ontology. A management component observes the ontology and maps newly added facts to management operations to adjust the system.

There are no other methods known to the authors for the combination of ontologies and Bayesian networks in an IT management context, but there are approaches to embed probabilistics into OWL. In [8] the embedding of probabilistic knowledge for OWL class membership is presented. The major problems are the representation of probabilistic knowledge in OWL, the derivation of an acyclic graph and the construction of conditional probability tables. Therefore, special OWL classes are defined to represent the expressions $P(A)$, $P(A|B)$ and $P(A|\overline{B})$, which have properties for conditions, values and probabilities. These properties are used to generate the conditional probability tables. A specially modified reasoner is needed to evaluate the ontology, as the existing reasoners cannot be used.

One problem that has to be taken into account when updating facts in a knowledge base, is that the knowledge base may enter an inconsistent state because of previously derived facts contradicting the changes. This is known as *belief change*, and in the context of ontologies, as *ontology change*. Several works approach this problem, e.g. [9], where the authors examine the applicability of solutions from belief change theory to ontologies. Another approach to the problem is taken in [10], which proposes an ontology update framework where ontology update specifications describe certain change patterns that can be performed.

## III. Architecture

A new architecture for ontology-based automated IT management is currently under development by the authors and the main ideas are sketched in this section. The architecture consists of a set of components (shown in Figure 1), which can be grouped into

- Importers that add new data to the ontology
- Reasoning components, which use the existing data to derive new knowledge
- Management components, which interact with the system under management.

The central element of the system is an ontology that is used as a shared knowledge base (blackboard) for all components. Each component can read data from the knowledge base and add or remove facts from it. Services are used for the inter-component communication. The architecture is designed to be used in a distributed fashion.

### A. Importers

The combination of different domain models raises the requirement for corresponding importers. These specific components know how to map the domain specific model to an ontology model. Hence, an interface is defined, which allows the use of new domain specific model importers. Implemented model importers are an ontology importer and a CIM importer. The ontology importer simply reads the data from an OWL ontology and adds the facts to the shared knowledge base. The CIM importer uses the mapping rules described in [3] to map the CIM schema to OWL facts.

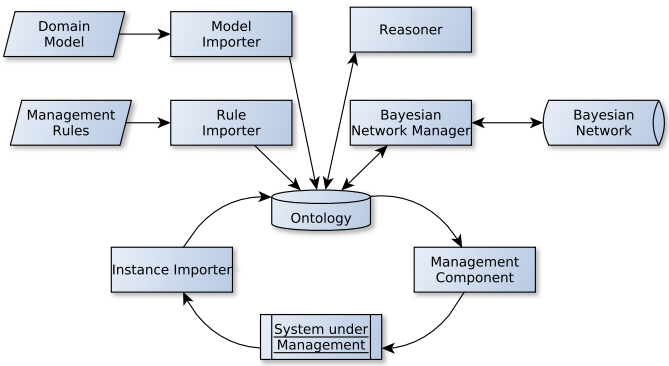


Figure 1. Components of the developed architecture

As well as models, rules can be specified in a domain specific manner. Hence, an interface is provided for the implementation of domain specific rule importers. Internally, SWRL is used as rule format for the shared ontology and an according importer was implemented.

In general, the domain model contains just the taxonomy of the monitored system but not the instance data. Therefore, a component is needed that monitors the system under management and imports runtime data into the ontology by creating according instances. Such components are called instance importers. An interface is provided for the integration of domain specific instance importers. Already implemented instance importers are the log record importer, which maps log records to instances and relations, and the CIM instance importer, which uses the OpenPegasus CIMOM to get information from a CIM-based management

 ISBN: 978-1-61208-175-5 

system. Other application-specific instance importers can be added as needed.

### B. Reasoning

The strength of OWL and its formal grounding is the ability to reason new knowledge from an existing knowledge base. In our architecture this feature is used to derive new facts from the domain specific models, the imported rules and the monitored instance data.

In many cases it is insufficient to just consider exact knowledge in IT management, because side effects and complex relationships are either not known or can not be modeled in an according abstraction. But especially for state prediction and root cause analysis probabilistic knowledge and the statistical consideration of historical data is needed. Because of that, a concept is used to make probabilistic modeling and reasoning possible, which is described in detail in [11]. The structure of the Bayesian network is derived from the OWL model. Specially annotated OWL instances become nodes and specially annoted OWL properties become arcs in the Bayesian model. The joint distribution tables are not modeled in the ontology directly, but trained using a maximum likelihood algorithm during a precedent training phase.

Ontologies are able to represent continuous and discrete variables, in OWL this is done using data properties. As Bayesian networks only work on discrete random variables, a discretization must be applied. To discretize continuous variables, some additional information is needed. OWL does not support the addition of supplemental data to data property assertions. Hence, a special variable class is defined, which has a data property that contains the actual value of the variable. There are three different types of variables: Continuous variables, Discrete variables and Enumerations. A mechanism is needed to map values of all three types of variables from the ontology to the generated Bayesian network and back again. Since enumerations generally have just a small state space, the values can be mapped one by one. For continuous and discrete variables the mapping is problematic and a discretization must be applied.

Because causal relationships can be seen as unidirectional edges between entities, the OWL object property concept can be used for their representation. In general it is not possible to connect data properties in OWL, but in this case it is feasible because all variables are already encapsulated by instances of the variable class.

For the evaluation of these relationships, causations are mapped to a Bayesian network where each instance of the variable class becomes a node. For numerical variables each variable is checked for intervals. A discrete state is created for each interval in the state space of the node in the network. Enumerations are checked for their defined enumeration class and for each individual of this class a state is created with the unique name of the individual. Causal relationships between variables become arcs in the Bayesian network.

In the next step the OWL model is analyzed for variable states, which will be set as evidences in the Bayesian network. Subsequently, an inference algorithm is applied to calculate the belief for the states of unobserved variables (variables which have no value set in the ontology). If the calculated belief is above a defined threshold, the deduced value is set for the variable in the ontology and can thereby be used by the exact reasoners for further reasoning. To ensure the knowledge exchange between the reasoning components a component can be called multiple times in a reasoning cycle.

### C. Management components

Management components are used to reconfigure the system under management. They contain the knowledge that is needed to interact with a specific component of the system. Depending on the evaluation results of the rules, according actions are triggered. When CIM is used as a domain model, the management components can call methods on the CIMOM, which in turn controls the particular component, or it can execute external commands directly.

### D. Runtime

The first step on application startup is the import of required domain models and rules using the according model and rule importers. After that, the management cycle is started (also known as MAPE-K loop [12], which stands for monitor, analyze, plan, execute and knowledge). The loop begins with the monitoring phase, where information from the system under management is read and imported into the ontology as instances.

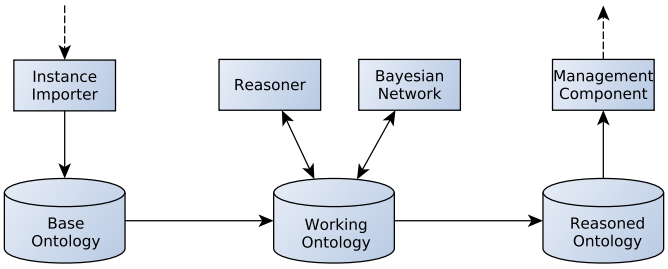


Figure 2. Multi step ontology reasoning process

In the analysis phase, the domain models, the rules and the monitored data are used for the reasoning of new knowledge. The reasoning process is shown in Figure 2.

The base ontology contains all the imported and monitored data. When the reasoning process starts, all data of the base ontology is copied into the working ontology. All reasoners are applied to this ontology sequentially and add their reasoned knowledge to it. When all reasoners have finished, the data of the working ontology is copied to

 ISBN: 978-1-61208-175-5

the reasoned ontology, which is used for queries into the knowledge base and stays untouched until the next reasoning phase has finished.

The reasoning takes place in this multi-step process for two reasons: The first reason is handling ontology change, as new information can be added easily to an ontology, but not retracted easily. By keeping the base model and inferred knowledge from different reasoners in separate sub-ontologies, inferred knowledge from a single reasoner can be retracted without effort. The second reason is that the last version of the *reasoned ontology* can still be queried, while the new version is being created. As reasoning can be slow on large ontologies, this makes sure that clients do not block on queries but can always receive an instant reply. The query result therefore may be as old as one reasoning cycle.

The last steps in the cycle are the plan and execute phases. The management components use the data of the reasoned ontology to make management decisions and execute them on the system under management. The presented architecture is partially implemented in Java using the OSGi Framework as service middleware. For the service abstraction the interfaces of the OWL API are used.

## IV. Conclusion

In this paper we sketched an approach for ontology-based IT management. An architecture that uses an ontology combined of the domain model, rules and dynamically updated instance data was presented. Two main problems must be solved: The first problem is the creation of a suitable domain model, which was covered by the translation of CIM to OWL and the expression of probabilistic knowledge using Bayesian networks. The integration of other domain models has yet to be examined. The second problem is the continuous update of the ontology with new facts. This is a topic of current research, and our solution is a multi-step reasoning process. Performance comparisons to other approaches and with different ontologies must be conducted.

Future work includes the development of importers for other domain models. It also includes the application of the developed tool on storage management and the ambient assisted living (AAL) context. Furthermore, performance needs to be optimized.

In the context of storage management the Storage Management Initiative Specification (SMI-S), which is a specialization of the CIM Model, can be used to manage storage systems. Rules, which are verbally defined in the specification, can be formalized and integrated into the OWL model. Besides, the probabilistic part can be used to make assertions about future states (e.g. how high is the probability of a full file system tomorrow if there is a peak) or to analyze previous scenarios (e.g. what was the most likely reason for a file server crash). In combination a pro-active management can be achieved and systems can be reconfigured before an error occurs.

In the context of ambient assisted living the domain will be a living environment, equipped with a set of sensors and effectors. That environment will be modeled in a hierarchy of ontologies and monitored during runtime. The observed data is used to derive higher level knowledge, e.g. that an elderly person lies on the ground and needs help.

## References

[1] Distributed Management Task Force, "Common Information Model (CIM)," http://www.dmtf.org/standards/cim/.

[2] World Wide Web Consortium, "OWL Web Ontology Language," http://www.w3.org/TR/owl2-overview/.

[3] A. Textor, J. Stynes, and R. Kroeger, "Transformation of the Common Information Model to OWL," in *10th International Conference on Web Engineering - ICWE 2010 Workshops*, ser. LNCS, vol. 6385. Springer Verlag, July 2010, pp. 163–174.

[4] J. E. L. De Vergara, A. Guerrero, V. A. Villagrá, and J. Berrocal, "Ontology-Based Network Management: Study Cases and Lessons Learned," *Journal of Network and Systems Management*, vol. 17, no. 3, pp. 234–254, 2009.

[5] A. Guerrero, V. A. Villagrá, J. E. L. de Vergara, A. Sánchez-Macián, and J. Berrocal, "Ontology-Based Policy Refinement Using SWRL Rules for Management Information Definitions in OWL," *Large Scale Management of Distributed Systems*, vol. 4269, pp. 227–232, 2006.

[6] J. E. L. De Vergara, V. A. Villagrá, and J. Berrocal, "Applying the Web ontology language to management information definitions," *IEEE Communications Magazine*, vol. 42, no. 7, pp. 68–74, July 2004.

[7] A. Westerinen, J. Schnizlein, J. Strassner, M. Scherling, B. Quinn, S. Herzog, A. Huynh, M. Carlson, J. Perry, and S. Waldbusser, "Terminology for policy-based management," United States, 2001.

[8] Z. Ding and Y. Peng, "A probabilistic extension to ontology language owl," in *In Proceedings of the 37th Hawaii International Conference On System Sciences (HICSS-37), Big Island*, 2004.

[9] G. Qi and F. Yang, "A survey of revision approaches in description logics," in *Web Reasoning and Rule Systems*, ser. LNCS, vol. 5341. Springer, 2008, pp. 74–88.

[10] U. Lösch, S. Rudolph, D. Vrandečić, and R. Studer, "Tempus fugit - towards an ontology update language," in *6th European Semantic Web Conference (ESWC 09)*, vol. 1. Springer, January 2009, pp. 278–292.

[11] F. Meyer, "Kombination von Modellen zur Systemanalyse im Selbstmanagement-Kontext," in *Workshop Self-Organising, Adaptive, Context-Sensitive Distributed Systems (SAKS 2011)*, ser. Electronic Communications of the EASST, March 2011.

[12] IBM Corporation, "An Architectural Blueprint for Autonomic Computing, Technical Whitepaper (Fourth Edition)," June 2006.

 ISBN: 978-1-61208-175-5 