A Knowledge-based Security Policy Framework for Business Process Management

Dong Huang
Siemens AG, Corporate Technology
Otto-Hahn-Ring 6
81739 Munich, Germany
dong.huang.ext@siemens.com

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

Abstract

Business Process Management (BPM) is often a key component of the business change. Business rules, whether embedded within BPM or on their own, have begun playing an ever-increasing role of prominence in process-centric business and system strategies. Service-oriented architecture (SOA) is used to support the business processes and a novel approach to share service knowledge and application-specific information is needed. In this paper, we model web service policy with corporate knowledge, which is defined as the amount of knowledge provided by individual agents. The proposal builds upon the project AKT's[1] *work in defining a Semantic Web Constraint Interchange Format (CIF), which itself builds on the proposed Semantic Web Rule Language (SWRL). The main contribution include also a new ontology for representing security constraints as policy and a knowledge management method to our proposed knowledge-based policy framework; we also show the possibility to integrate the business rules into policy specification by means of converting them into Constraint Satisfaction Problem (CSP) using CIF.*

1. Introduction

Business Process Management (BPM) is often a key component of the business change. Business rules, whether embedded within BPM or on their own, have begun playing an ever-increasing role of prominence in process-centric business and system strategies. Service-Oriented Architecture (SOA) is used to support the business processes and such a Service-Oriented Architecture describes principles for creating dynamic, loosely coupled systems based on services, but no single specific implementation. The concept of Web Services is thought to be the next generation of e-business architectures for the web. A novel approach to share service knowledge and application-specific information is needed.

A security policy framework for business processes was proposed in [12], which gave a preliminary idea of policy management and implementation in SOA. In a new line of work, this framework is further extended and a knowledge-based policy framework for service-oriented computing (SOC) is introduced.

Agent Oriented Abstraction (AOA) [5] addresses knowledge management issues by viewing an e-business services-oriented network as a multi-agent paradigm for distributed computation. The concept of a *virtual knowledge community (VKC)* enables us to model *corporate knowledge* as the amount of knowledge provided by individual agents [16]. In our approach, corporate knowledge can provide the basic information model for policy specification and conflict resolution by capturing business rules and the non-functional requirements of web services.

The idea in our approach is to gather pertinent data/knowledge from multiple stakeholders in the e-business scenario, along with constraints specified by non-functional requirements of web services and business rules. These data and constraints are then fused by mediator software into a dynamically composed Constraint Satisfaction Problem (CSP), which is then dispatched to a solver inside the knowledge-based policy framework. The security constraints are expressed against a semantic data model/ontology because it may be necessary to transform them at run-time. Security constraints in our approach are represented using an expressive quantified constraint language, the Constraint Interchange Format (CIF) [19].

The remaining sections of this paper are structured as follows. Section 2 scratches the surface of the agent approach for web services and introduces the principle of AOA and VKC. Section 3 outlines the requirements of the security constraints specification language for web services

[1] http://www.csd.abdn.ac.uk/research/akt/

and introduces the principle of Constraint Interchange Format. An overview of our knowledge-based policy framework for web services is also given in this section. Section 4 surveys related works. Section 5 includes the future research direction for the work and conclusion.

2 Agents, AOA and VKC

This section outlines several different, although interrelated, agent-based viewpoints for web services, agent abstraction and corporate knowledge.

2.1 Agents and Web Services

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 for their users [10]. Uniting agents and web services can enhance the construction and flexibility of web service applications [24].

Before introducing other agent-based approaches in following sections, it is necessary to remind some agent related definitions first. An agent is capable of autonomous action in assigned environment in order to meet its design objectives [27]. 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 mainly used to better formalize problems in Distributed Artificial Intelligence (DAI) [26].

The world of web services is characterized as loosely-coupled distributed systems based on SOC. The use of web services could be considered as actions that the agent might execute to meet its goals. In [14] four major trends in internet computing were analyzed which have driven SOC and Multi Agent Systems (MAS) research into the future. They are among emerging approaches with MAS-like characteristics in SOC, such as ubiquitous computing, ontologies, service-level agreements and quality-of-service measures for instance. All of them can be suitably tackled with MAS concepts and techniques.

2.2 Agent Oriented Abstraction

The agent oriented abstraction paradigm is introduced in [5]. It is a high level abstraction for agent modeling and covers the concepts of agents, annotated knowledge, utility functions and society of agents. Indeed, AOA relies on Weber's classical theory in Sociology [25].

In the approach of AOA, agents are seen as objects and through their knowledge contents, they are organized into annotations that gather classes. Encapsulation, inheritance and polymorphism are features that can be adequately defined. The AOA model can be abstractly summarized by a number of basic definitions. A detailed instruction of that will not be given here, but is to be found in [5].

Chiefly, in AOA all agents have two parts: a decision mechanism and knowledge. For the former, a scope of possible classification of utility was given: expected utility function, the common sense measure of usefulness, the class of models and the class arising from logical modeling. For the latter, there are also some associated classes: ontology, communication, cognition and safety. Based on the knowledge annotations, agents can generate utility related to its tasks and goals.

Within the AOA approach web services and their related policies can be abstractly modeled in the knowledge part of agents running around the semantic web.

2.3 Virtual Knowledge Communities

In [16] the application of the AOA model to the abstract modeling of corporate knowledge is investigated. To avoid the separation between agents and knowledge, it was considered that agents have explicitly represented knowledge and communication ability.

Traditionally, information is mostly centralized within a uniform information structure. This view point is not truly compliant with the nature of knowledge that is subjective, distributed and contextual [3]. From the perspective of the knowledge information society, modern knowledge management often focuses on the constitution of communities of practice and communities of interest [9].

The concept of a community of practice or a community of interest can be supported in a virtual community in order to bring the concerned agents together to share their knowledge with each other. A community is a place where agents can meet and share knowledge with other agents which share a similar domain of interest. The concept of a VKC was introduced as a means for agents to share knowledge about a topic [17]. It aims to increase the efficiency with which information is made available throughout the society of agents.

From the point of view of corporate knowledge management, agents can be individuals, software assistants or automata. Agents possess knowledge and processes within the society tend to make agents produce and exchange knowledge with each other. These processes are distributed throughout the society and contribute through their own intrinsic goals to solve a unique high-level challenge. This provides the link between corporate knowledge and VKC [16].

Community modeling has some key notions: domain of interest, community pack, community buffer. A domain of interest exists in each VKC and is similar to the concept of ontology for an agent. It is given by the community leader which created the community. The community pack is what defines the community. It consists of a community knowledge cluster, a normalized ontology which contains at least the head of the community cluster, and the identification of the leaders of the community. The community buffer can record messages which are used by the member of a community to share their knowledge. This approach is compatible with blackboard systems, but still has its difference, because agents cooperate to solve their respective problems, not for a unique goal.

The VKC approach has been designed and partially implemented as a prototype system. The implementation is based on Java Agent Development Framework (JADE) and Java Runtime Environment (JRE) platform. It was tested and evaluated. A component of the system enables us to simulate virtual knowledge communities (VKCs).

3 Security Policy Framework

Security policy can mean different things at different times. In this paper, security would involve ensuring access control, confidentiality, integrity.

3.1 Security Constraints Specification Language

Various approaches have been done to achieve security constraints specifications, including logic-based languages, role-based access control, various access controls and trust specification techniques [6]. But, a specification language, which can meet the following requirements, is still missing.

- **Support of non-functional service descriptions.** How to model non-functional properties into the policies and enable reasoning over them?

- **Integration of Business Rules.** Business rules state core business policies. They control and influence business behavior. How to integrate them with the knowledge base and specify policy using rules?

Various web services and semantic web services approaches such as UDDI[2], OWL-S[3], SWSF [2] and WSMO/WSML [8] have been investigated to describe the non-functional properties of a service. In [21] a set of the most relevant non-functional properties for Web services and their modeling are described. An overview of all these approaches is given in [23].

Business Rules are used for categorizing facts important to a business. They also require or prohibit actions by a business. OMG[4] has been widening its scope to include business modeling. Several of its recent requests for proposals have been about or related to business rules. These proposals are Semantics of Business Vocabulary and Business Rules, Production Rule Representation, Business Rule Management.

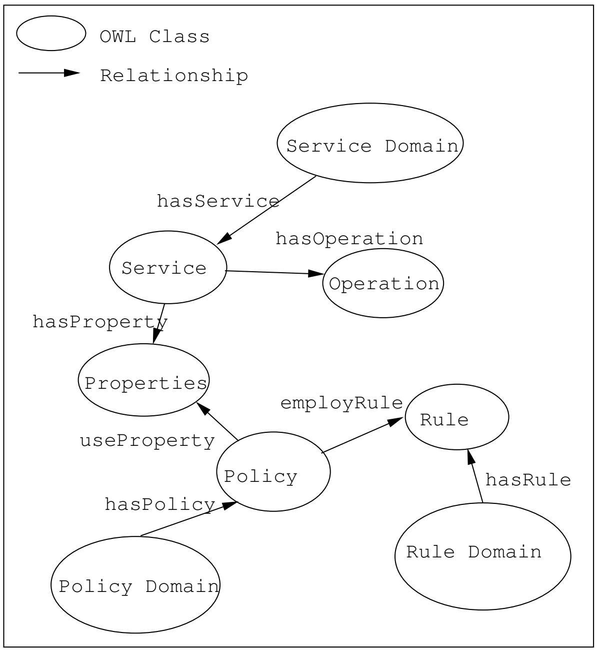

Figure 1. Upper ontology for policy

We define the constraints specification language with an upper ontology in Figure 1.

- **Service Domain** is a collection of services. The service types can be composite service and atomic service. The services may be chained together to a composite service for special business goals or processes.

- **Policy Domain** categorizes the policies by different aspects like security, trust and management. Meta-Policy, so called policy of policy, can be defined in this domain.

- **Rule Domain** aims to represent the requirements of business activities, which may be application-specific terms, e.g. Legal Rules applied to online-shopping.

- **Properties** apply for all service descriptions: functional, behavioral and non-functional.

- **Rule** is a statement that can be represented as IF *Condition* THEN *Action*.

[2] http://www.uddi.org

[3] www.daml.org/services/owl-s

[4] http://www.omg.org

From the upper ontology, a domain-specific ontology describes the vocabularies, business rule terms, service descriptions used within the domain. By using the domain-specific specification template, a constraints specification can be generated and exchanged automatically with Ontology language OWL [18], Rules language SWRL [11] and CIF [19], which is described in the next section.

3.2 Constraint Interchange Format

Constraint Interchange Format (CIF) is based on the Colan [1] constraint language, which is based on range restricted first order logic (FOL). Earlier versions of the language were aligned with Resource Description Framework (RDF) [15] and SWRL. CIF constraints are essentially defined as quantified implications, so we re-use the implication structure from SWRL, but allow for nested quantified implications within the consequent of an implication. An example CIF constraint is shown in human-readable SWRL-style syntax below:

$$\begin{gathered}(\forall ?x \in X, ?y \in Y) p(?x, ?y) \wedge Q(?x) \Rightarrow \\ (\exists ?z \in Z) q(?x, ?z) \wedge R(?z) \Rightarrow \\ (\forall ?v \in V) s(?y, ?v)\end{gathered}$$

In [19], an RDF/XML syntax is provided as an extension to the one given for SWRL to support publishing and interchange of CIF constraints. A new **rdfs:Class** Constraint, with properties hasQuantifiers and hasImplication is defined. For example, if we wanted to introduce a business requirement like “every delegation group must contain at least one participant from government”, the following code shows RDF/XML for this constraint.

```
<cif:Constraint>
 <cif:hasQuantifiers
      rdf:parseType="Collection">
  <cif:Forall>
   <cif:var rdf:resource="#g"/>
   <cif:set rdf:resource="#Delegationgroup"/>
  </cif:Forall>
  <cif:Exists>
   <cif:var rdf:resource="#p"/>
   <cif:set rdf:resource="#Government"/>
  </cif:Exists>
 </cif:hasQuantifiers>
 <cif:hasImplication>
   <swrl:Imp>
    <swrl:body rdf:parseType="Collection"/>
    <swrl:head rdf:parseType="Collection">
     <swrl:IndividualPropertyAtom>
      <swrl:classPredicate
            rdf:resource="#has-member"/>
      <swrl:argument1 rdf:resource="#g"/>
      <swrl:argument2 rdf:resource="#p"/>
     </swrl:IndividualPropertyAtom>
    </swrl:head>
   </swrl:Imp>
 </cif:hasImplication>
</cif:Constraint>
```

RDF/XML for the constraints

Rule Interchange Format (RIF)[5] is another interchange formats for logic expressions on the Web. It is expected that CIF will evolve to use RIF in place of SWRL as the new format takes shape. As it is currently planed, Phase 1 RIF is essentially Horn Logic. If Phase 2 RIF includes full FOL then this format may wholly subsume CIF. At that point it is conceivable to simply define CIF as a subset of RIF: constraints would be interchanged in RIF itself [22].

3.3 Framework Architecture

In order to support conflict resolution and life-cycle management of policies, as well as enable reasoning over the knowledge base, an architecture of the knowledge-based policy framework is illustrated in Figure 2. It includes two supporting services: a policy service and a knowledge service. Additional components include: reasoner, management tools and repository.

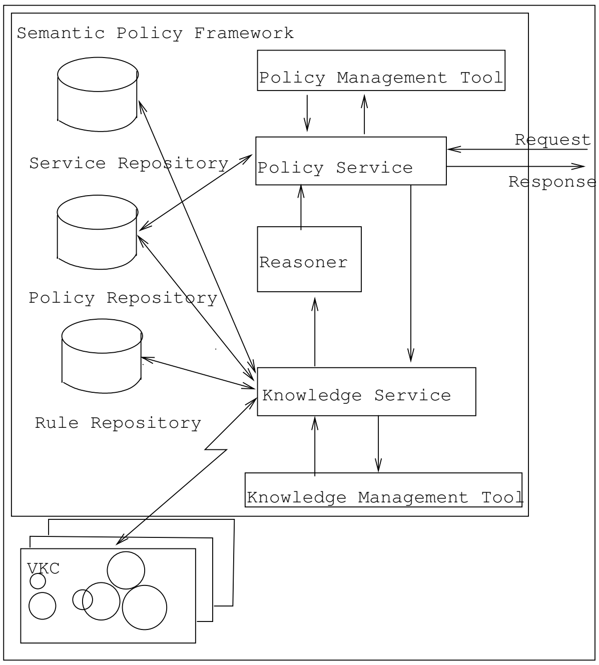

Figure 2. Framework architecture

- **Policy Service.** The policy service acts as a Policy Decision Point (PDP)[6], for web services policies, including security and QoS requests. The policy service acts on service requests and renders a decision.
- **Knowledge Service.** The knowledge service manages the VKCs and the repository of static knowledge base: business rules, policies and service descriptions.

[5] http://www.w3.org/2005/rules/

[6] http://www.ietf.org/

- **Repositories.** Three repositories are used to store service descriptions, policies and business rules. This information can be managed through the policy and knowledge service.

- **Reasoner.** Reasoner is used to perform logical inference over corporate knowledge based on the knowledge repositories and VKCs. In our implementation, we use KAON2 [7] as reasoner. The major advantage of KAON2 is that it is a very efficient reasoner when it comes to reasoning with Description Logics ontologies containing very large ABoxes and small TBoxes [20]. The terms Abox and Tbox are used to describe two different types of statements in ontologies. Together Abox and Tbox statements make up a knowledge base.

- **Management Tools.** The policy management tool acts as the interface to policy service; it manages the lifecycle of policies, creates and deploys new policies. The knowledge management tool provides an interface to manage the VKCs and knowledge repositories.

VKCs build a special knowledge component outside the framework and are managed by the Knowledge Service Agent, which also acts as a leader agent on the knowledge service. As the policy service acts as a PDP, it receives the service request from the Policy Enforcement Point (PEP) at the service end. There are two possible options: a) It finds out the suitable policies from the policy repository, renders the decision and sends the result back to PEP, b) It cannot find the proper policy or there are conflicts in policies, the request will be converted to a request to the knowledge service. The knowledge service will analyze the request and prepare a knowledge base for the next reasoning step. The knowledge base is built both on static knowledge from repositories and dynamic knowledge from VKCs. The concept of knowledge integration is described in [13]. Based on the result of reasoning on the knowledge base, the policy service can make its decision of the service request.

4 Related Work

Various approaches have already been done in both industry and academia.

WS-Policy[8] defines a framework and a model for the expression of the capabilities, requirements, and general characteristics of entities in an XML Web Services-based system as policies. Policy expressions allow for both simple declarative assertions as well as more sophisticated conditional assertions. But the WS-Policy framework lacks formal semantics, it prevents us to determine its expressivity and computational properties.

In [7], a complete policy-based management framework is presented, which includes a policy specification language and architecture for deploying policies.

KAoS Policy and Domain Services [4] use ontology concepts encoded in OWL to build policies. These policies constrain allowable actions performed by actors which might be clients or agents. The applicability of the policy is defined by a class of situations whose definition can enclose components specifying required history, state and currently undertaken actions. Mandatory action can be annotated with different constraints restricting possibilities of its fulfillment.

All of these approaches did not address knowledge exchange and sharing among all the stakeholders in the e-business scenario.

Rein[9] is a decentralized framework for representing and reasoning over distributed policies in the Semantic Web. Rein (Rei and N3) uses high level Rei concepts for policies and N3 rules to connect these policies to each other and the Web. Policies in Rein use defined information and inferences made by other policies and web resources forming interconnected policy networks. Rein allows policies to be represented in different policy ontologies and uses N3 rules, a semantic web rule language, for defining the connections in these networks. Reasoning over these networks to obtain policy decisions is done using cwm, an N3 reasoner.

In our approach, the framework is somehow centralized, which is designed to support the special service platform and act as a policy management component.

5 Conclusion and Future Works

We have investigated a distributed knowledge management approach to help modeling web services policies. This approach is based on the concept of corporate knowledge through the use of VKCs. By integrating the knowledge management service, the knowledge-based policy framework is able to access external VKCs which can provide application-specific knowledge on transactions in e-business. After the knowledge integration, rich corporate knowledge can be used to fulfill the task of reasoning and enrich the policy with former unavailable information like non-functional requirement of service and business rules affected by the transaction processes. In this paper, we have also proposed a representation for security constraints at the Semantic Web logic layer. We illustrated the use of the CIF/SWRL constraints and new upper ontology to integrate the business rules and non-functional descriptions of web services in the policy specification.

The development of the knowledge-based policy frmework is ongoing; currently we are trying to employing different kinds of Reasoner to evaluate the complexity, scalability and performance.

[7] http://kaon2.semanticweb.org/

[8] www.ibm.com/developerworks/library/ws-polfram

[9] http://groups.csail.mit.edu/dig/2005/05/rein

The aim is to enable a knowledge-based policy framework and knowledge management methodology, which enable security, trust and QoS in service-oriented computing environments and provide a novel solution for fields like e-business, telecommunication and enterprise application integration.

Acknowledgments

One of the authors, Dong Huang, would like to thank the team at Siemens for their guidance and support whilst conducting this research. In particular, he thanks Jorge Cuellar for his valuable contributions towards this work.

References

[1] N. Bassiliades and P. G. CoLan. A functional constraint language and its implementation. *Data and Knowledge Engineering*, pages 203–249, 1994.

[2] S. Battle, A. Bernstein, H. Boley, M. Gruninger, and R. Hull. Semantic web services framework (SWSF) overview version 1.0. *Semantic Web Services Initiative (SWSI)*, 2005.

[3] M. Bonifacio, P. Bouquet, and R. Cuel. Knowledge nodes: the building blocks of a distributed approach to knowledge management. *Journal of Universal Computer Science*, 8(6):652–661, 2002.

[4] J. Bradshaw and A. Uszok. Representation and reasoning for daml-based policy and domain services in kaos and nomads. In *AAMAS '03: Proc. of the second international joint conference on Autonomous agents and multiagent systems*, pages 835–842, New York, NY, USA, 2003. ACM Press.

[5] 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.

[6] N. Damianou, A. Bandara, M. Sloman, and E. Lupu. A survey of policy specification approaches. Technical report, Department of Computing, Imperial College of Science Technology and Medicine, 2002.

[7] N. Damianou, N. Dulay, E. Lupu, and M. Sloman. Ponder: A language for specifying security and management policies for distributed systems. Technical report, Imperial College, October 2000.

[8] D. Fensel and C. Bussler. The web service modeling framework WSMF. Technical report, Vrije Universiteit Amsterdam, 2002.

[9] G. Fischer and J. Ostwald. Knowledge management: Problems, promises, realities, and challenges. *IEEE Intelligent Systems*, 16(1):60–72, 2001.

[10] J. Hendler. Agents and the semantic web. *Intelligent Systems*, 21(2):30–37, 2001.

[11] I. Horrocks and P. F. Patel-Schneider. SWRL: A semantic web rule language combining OWL and RuleML. Technical report, The Rule Markup Initiative, May 2004.

[12] D. Huang. Semantic policy-based security framework for business processes. In *Proc. of the Semantic Web and Policy Workshop*, Galway, Ireland, November 2005.

[13] D. Huang, Y. Yang, and J. Calmet. Modeling web services policy with corporate knowledge. In *Proc. of 2006 IEEE International Conference on e-Business Engineering*, Shanghai, China, October 2006.

[14] 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.

[15] K. Hui, S. Chalmers, P. Gray, and A. Preece. Experience in using rdf in agent-mediated knowledge architectures. *Agent-Mediated Knowledge Management (LNAI 2926). Springer-Verlag*, pages 177–192, 2004.

[16] P. Maret and J. Calmet. Modeling corporate knowledge within the agent oriented abstraction. In *Proc. of International Conference on Cyberworlds (CW'04)*, pages 224–231. IEEE Computer Society, 2004.

[17] P. Maret, M. Hammond, and J. Calmet. Virtual knowledge communities for corporate knowledge issues. In *Proc. of 5th International Workshop on Engineering Societies in the Agents World (ESAW'04)*, volume 3451 of *Lecture Notes in Computer Science*, pages 33–44, Toulouse, France, October 22-24 2004. Springer.

[18] D. L. McGuinness and F. van Harmelen. Web ontology language overview. Technical report, W3C Recommendation, February 2004.

[19] C. McKenzie, P. Gray, and A. Preece. Extending SWRL to express fully-quantified constraints. In *Proc. of RuleML 2004 Workshop at ISWC 2004*, Hiroshima, Japan, November 2004.

[20] B. Motik, U. Sattler, and R. Studer. Query answering for OWL-DL with rules. In *Proc. of International Semantic Web Conference 2004*, pages 549–563, Hiroshima, Japan, November 2004.

[21] J. O'Sullivan, D. Edmond, and A. H. M. ter Hofstede. Formal description of non-functional service properties, queensland university of technology. Technical report, http://www.service-description.com, 2005.

[22] A. Preece, S. Chalmers, C. McKenzie, J. Pan, and P. Gray. Handling soft constraints in the semantic web architecture. In *Proc. of RoW2006 Reasoning on the Web at WWW2006*, Edinburgh, UK, 2006.

[23] I. Toma and D. Foxvog. Non-functional properties in web services. Technical report, DERI, 2006.

[24] C. D. Walton. Uniting agents and web services. *AgentLink News*, (18):26–28, August 2005.

[25] M. Weber. *Economy and society*. University of California Press, 1986.

[26] G. Weiss. *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. MIT Press, 1999.

[27] 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.