# Integration of KAoS Policy Services with Semantic Web Services

**Andrzej Uszok, Jeffrey M. Bradshaw, Gianluca Tonti, Renia Jeffers**
*Institute for Human and Machine Cognition (IHMC), 40 S. Alcaniz, Pensacola, FL 32502, USA*
*{auszok, jbradshaw, gtonti, rjeffers}@ihmc.us*

**Lars E. Olson**
*The University of Illinois, Urbana-Champaign, IL 61801, USA*
*leolson1@uiuc.edu*

## 1. Introduction

In this demo we present the integration of the universal ontology-based KAoS Policy and Domain Services [2, 10] with Semantic Web Services [1, 5, 8] running inside the Tomcat servlet container (*http://jakarta.apache.org/tomcat*).

KAoS has pioneered the use of Semantic Web language—in this case OWL—to represent policy[1]. KAoS services have been extended to work equally well with both agent-based (e.g., CoABS Grid, Cougaar, SFX, Brahms) and traditional clients on a variety of general distributed computing platforms (e.g., CORBA, Grid Computing (Globus GT3)). KAoS uses ontology concepts encoded in OWL to build policies (see *http://ontology.ihmc.us/C-Map/Policy.html*).

Policies, which constrain the behavior of system components, are moving towards the center of interest of the Web service community[2]. There are many opportunities for policy application within Web Services (see link in footnote 2, [4, 6]). In this demo we will show how KAoS was adapted to express and enforce policies on the use and behavior of the Web Services, described using OWL-S and deployed in Tomcat. We will show how concepts from OWL-S descriptions are used to create policies and what necessary components are introduced into Tomcat to allow for policy enforcement.

Additionally, we will show our integration with TrustBuilder[3], which is used for trust negotiation between a client and a service. This is, however, the first step towards the full integration as TrustBuilder currently is using its own syntax for policy expression. We are working towards the usage of KAoS to express the TrustBuilder policies, as well.

The demo will be illustrated using OWL-S services defined for the CoSAR-TS[4] (Coalition Search and Rescue Task Support) project. The services include rescue resources, medical facilities, notification mechanism, and so forth, which are constrained by different policies on how they may be used in a particular coalition context.

## 2. Demo Content

The demo will consist of two parts. First the generic architecture and features of KAoS will be presented including, among other components, KPAT, the KAoS GUI. Following this, KAoS integration with Tomcat and the use of OWL-S ontologies to construct policies will be demonstrated.

### *2.1 KAoS Policy Service Demonstration*

In the scope of KAoS presentation we will show the following.

<u>KAoS Bootstrap and Configuration</u>: During its bootstrap, KAoS first loads a KAoS Policy Ontology (KPO) defining concepts used to describe a generic actors' environment and policies within this context (*http://ontology.ihmc.us/*). Alternatively, a previously saved configuration containing namespaces, policies, etc. can be loaded.
<u>Ontology Namespace Browsing and Management:</u> KPAT allows for browsing of loaded ontologies: examining their content; classes, properties, instances and imported namespaces. It also allows dynamically adding new ontologies on the fly extending concepts from the generic ontology, with notions specific to the particular controlled environment.
<u>Domain and Actor Class creation</u>: KAoS allows for different ways of expressing the subject of the policy, either through explicit domain membership or implicitly by values of properties defined as ontological actor classes. Dynamic creation of concepts for both these entities will be presented.
<u>Policy Creation</u>**:** Illustrative examples of policies will be created using KPAT. The tool guides a user through a creation process using ontology defined ranges to always narrow user choices to the most appropriate set of values; only these valid in the given context. The OWL encoding of the created policy will be presented.
<u>Policy Distribution</u>: After a policy is created it has to be distributed, through the KAoS Directory Service, to Guards, which are policy decision points located close to the running entities; usually one per a Java VM. In the process of policy distribution first the description logic classification algorithm is use to find out if given policy should be distributed to a particular Guard controlling given actors (KAoS has to determine if they potentially fall into the scope of the given policy). Then the policy has to be translated from the OWL format to more efficient form and the necessary subsumption results on its concepts have to be cached into the policy packaged send to the Guard in order to make the policy decision process efficient.

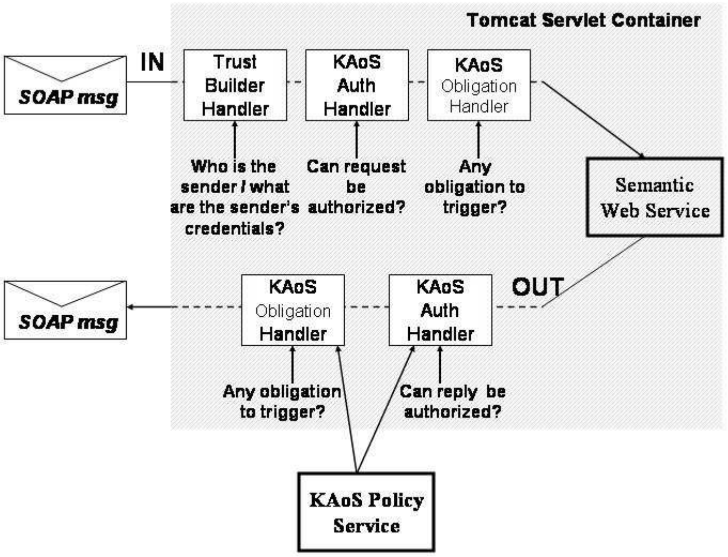
[1] A comparison among two semantically-rich representations of policy (KAoS, Rei) and amore traditional policy language (Ponder[3]) can be found in [9].
[2] http://www.w3.org/2004/06/ws-cc-cfp.html
[3] http://isrl.cs.byu.edu/TrustBuilder.html
[4] http://www.aiai.ed.ac.uk/project/cosar-ts/

Policy Disclosure**:** When many policies have been defined, policy disclosure (i.e., finding out which policies apply to a given situation) becomes complex. Like policy distribution, policy disclosure exploits the description logic classification algorithm. KAoS functionality in this area is not limited to the determination if a given action is authorized or forbidden or if it triggers some obligation but also allows for exploration of policy options. For instance; it is possible to find for a given partially specify action what values are allowed for its given property. The appropriate API is available to be used in the code. In addition, KPAT contains a graphical interface that allows building an action definition hypothetically performed by a given actor and also allows testing how this action would be affected by the policies. This GUI will be presented in addition to a running demo of agents, whose actions will be constrained by policies.

Policy Analyses**:** Policies being introduced by many users and at different time can and usually are involved in unexpected and unintended conflicts or overlaps. KAoS allows to analyze policies' interactions and, if necessary, to modify them. The description logic subsumption reasoning is used to find relations between classes of situations and actions controlled by different policies in order to determine if they are in conflict. The KPAT graphical interface for this functionality will be presented with example conflicting and overlapping policies.

## *2. KAoS Semantic Web Services integration*

The figure below presents elements controlling a SOAP request before it reach its target Semantic Web Service running as a servlet within Tomcat. Likewise, the SOAP response is similarly controlled.



The first handler on the way of SOAP request entering Tomcat is the TrustBuilder handler which negotiates initial credentials of a client and registers the client within the local KAoS in specified domains and with negotiated actor classes. This will allow KAoS to classify the client and apply appropriate policies to its actions. This handler can also renegotiate the trust with the client in case some of its requests are rejected. As an effect the client registration within KAoS changes so different policies will apply to its actions.

The next are two KAoS handlers that use WSDL information about the called method in the SOAP message to find out in the annotated WSDL file available through the included in the message URL the reference to the OWL-S ontology defining this service. This allows building the ontologically annotated description of the request and asking KAoS about its policy decision regarding this request. First, the authorization handler checks if the given request is allowed. Then if request passed the authorization handler, the obligation handler checks if the request triggers some additional obligations. It is assumed that these obligations refer to some Web Services which will be called by the handler. For instance a policy may require consultation or registration of performed transactions in some logging service available on the Web.

We will present this functionality by changing access and obligations on different OWL-S defined services from the CoSAR-TS project, see: http*://ontology.ihmc.us/CoSAR-TS/Demos/CoSAR-TS_Demo_Concept.htm*.

## 3. Software Availability

The web site: *http://ontology.ihmc.us/* contains the OWL ontology used by KAoS Policy Service.

The KAoS GUI – KPAT can be currently run from the Web using the Java Web Start technology from the following link: *http://norma.coginst.uwf.edu:8080/coalition/KPAT-TCP.jnlp*.

## Acknowledgements

This material is based on research sponsored by the Defense Advanced Research Projects Agency (DARPA) CoABS, DAML, and Ultra*og programs the US Air Force Research Laboratory under agreement numbers F30602-00-2-0577 and F30602-03-2-0014. Thanks to the other members of the KAoS project team: Maggie Breedy, Larry Bunch, Matthew Johnson, Hyuckchul Jung, Shri Kulkarni, James Lott and William Taysom. We are also grateful for the contributions of Austin Tate, Jeff Dalton, Mark Burstein, Pat Hayes, Luc Moreau, Niranjan Suri, Paul Feltovich, Richard Fikes, Jessica Jenkins, Bill Millaar, Deborah McGuinness, Rich Feiertag, Timothy Redmond, Rebecca Montanari, Sue Rho, Ken Ford, Mark Greaves, Jack Hansen, James Allen, and Robert Hoffman.

## References

[1] Ashri, R., Payne, T. R., & Surridge, M. (2004). Towards a Semantic Web Security Infrastructure. *AAAI Spring Symposium on Semantic Web Services*. Stanford University,

[2] Bradshaw, J. M., Uszok, A., Jeffers, R., Suri, N., Hayes, P., Burstein, M. H., Acquisti, A., Benyo, B., Breedy, M. R., Carvalho, M., Diller, D., Johnson, M., Kulkarni, S., Lott, J., Sierhuis, M., & Van Hoof, R. (2003). Representation and reasoning for DAML-based policy and domain services in KAoS and Nomads. *Proceedings of the Autonomous Agents and Multi-Agent Systems Conference (AAMAS 2003)*. Melbourne, Australia, New York, NY: ACM Press,

[3] Damianou, N., Dulay, N., Lupu, E. C., & Sloman, M. S. (2000). *Ponder: A Language for Specifying Security and Management Policies for Distributed Systems, Version 2.3*. Imperial College of Science, Technology and Medicine, Department of Computing, 20 October 2000.

[4] Denker, G., Kagal, L., Finin, T., Paolucci, M., & Sycara, K. (2003). Security for DAML Web Services: Annotation and Matchmaking. In D. Fensel, K. Sycara, & J. Mylopoulos (Ed.), *The Semantic Web—ISWC 2003. Proceedings of the Second International Semantic Web Conference, Sanibel Island, Florida, USA, October 2003, LNCS 2870.* (pp. 335-350). Berlin: Springer.

[5] Fensel, D., Hendler, J., Lieberman, H., & Wahlster, W. (Ed.). (2003). *Spinning the Semantic Web.* Cambridge, MA: The MIT Press.

[6] Kagal, L., Finin, T., & Joshi, A. (2003). A policy-based approach to security for the Semantic Web. In D. Fensel, K. Sycara, & J. Mylopoulos (Ed.), *The Semantic Web—ISWC 2003. Proceedings of the Second International Semantic Web Conference, Sanibel Island, Florida, USA, October 2003, LNCS 2870.* (pp. 402-418).: Springer.

[7] Li, N., Grosof, B. N., & Feigenbaum, J. (2003). Delegation logic: A logic-based approach to distributed authorization. *ACM Transactions on Information Systems Security (TISSEC)*, 1-42.

[8] McIlraith, S. A., Son, T. C., & Zeng, H. (2001). Semantic Web Services. *IEEE Intelligent Systems*, 46-53.

[9] Tonti, G., Bradshaw, J. M., Jeffers, R., Montanari, R., Suri, N., & Uszok, A. (2003). Semantic Web languages for policy representation and reasoning: A comparison of KAoS, Rei, and Ponder. In D. Fensel, K. Sycara, & J. Mylopoulos (Ed.), *The Semantic Web—ISWC 2003. Proceedings of the Second International Semantic Web Conference, Sanibel Island, USA, 2003, LNCS 2870.* (pp. 419-437). Berlin: Springer.

[10] Uszok, A., Bradshaw, J. M., Jeffers, R., Suri, N., Hayes, P., Breedy, M. R., Bunch, L., Johnson, M., Kulkarni, S., & Lott, J. (2003). KAoS policy and domain services: Toward a description-logic approach to policy representation, deconfliction, and enforcement. *Proceedings of Policy 2003*. Como, Italy..