A study in the expressiveness of semantically different policy modelling schemes

Christos Tsarouchis, Declan O'Sullivan, David Lewis
Knowledge & Data Engineering Group (KDEG)
Department of Computer Science & Statistics, Trinity College Dublin, Dublin, Ireland
{tsaroucc | Declan.OSullivan | Dave.Lewis}@cs.tcd.ie

Abstract— **Policy Engineering is the process of authoring IT management policies, detecting and resolving policy conflicts and revising existing policies to accommodate changing IT resources, business goals and business processes. Policy authoring is often followed by policy enforcement where the actions specified by subjects are performed on targets (resources). In this paper, we study the use of semantically enhanced techniques, such as ontologies, to model *resources* and their corresponding *actions,* coupled with a mechanism that can accommodate frequent organizational change, to model policy *subjects.* For the modeling of policy subjects, the rule-based Community-based Policy management will be used. This integration falls into the category of combining Description Logics (DL) and Logic Programs (LP). We aim to study this integration primarily from the scope of overall system expressivity, but also from the scope of minimizing the cognitive load perceived by policy authors. Such an evaluation can help determine shortfalls in the design of the software system or of the policy model used. To study the balance in modeling with DL and LP techniques, the encoding of part of the Trinity College Dublin statutes will be performed, which is a sufficiently complex real-world example.**

Keywords; IT policy engineering, policy modeling, DL and LP integration

I. INTRODUCTION

Policy Engineering is the process of authoring IT management policies, detecting and resolving policy conflicts and revising existing policies to accommodate changing IT resources, business goals and business processes[1].

Policy authoring is often followed by policy enforcement where the actions specified by subjects are performed on targets (resources). In this paper, we study the use of semantically enhanced techniques, such as ontologies, to model *resources* and their corresponding *actions,* coupled with a mechanism that can accommodate frequent organizational change, to model policy *subjects.* Other research initiatives have attempted to solve such problems by combining the efficiency and the expressive power of rule-based systems with the semantic richness found in Description Logics [2]. For instance, a rule-based system can express variables and causality relationships whereas ontologies on the other hand, can represent relationships between entities in a taxonomic manner that could simplify maintainability.

In this paper, for the modeling of policy subjects, the rule-based Community-based Policy management will be used. This

integration falls into the category of combining Description Logics (DL) [3] and Logic Programs (LP) [4]. We aim to study this integration primarily from the scope of overall system expressivity, as well as from the scope of minimizing the cognitive load [5] perceived by policy authors, which is particularly important in collaborative environments, with multiple policy authors.

II. BENEFITS OF USING SEMANTICALLY ENHANCED RESOURCES

One of the requirements in policy engineering is the need to have an accurate and up-to-date view of the managed *resources* as well as of the *actions* that can be performed on these resources. This is particularly challenging in the case when not only these resources are frequently changing but also when their corresponding actions change as well.

For instance, using the example shown in [6] , we assume that a corporation needs to keep a record of all assets they possess, such as network routers, servers, PCs, etc, as well as a list of countermeasures that need to be taken either when the security of those assets is compromised, or proactively, so that they won't be compromised. These countermeasures include software patches, malware scans, fine-grained control of firewall ports etc. It is worth pointing out that such an ecosystem could be implemented in devices with different architectures, with each one having different characteristics (e.g. different memory size). These devices can also be based on different operating systems, which could require a different set of actions during a software update. For example, one system might require install only, the other might require install and reboot.

Ideally, a policy author should be able to correctly enforce a policy of the following form: "Install software update X on all applicable devices". As the complexity of the managed resources is increasing, the enforcement of such a high level policy poses a very challenging problem. This policy would require its "refinement" into the appropriate low-level policies, something that often takes various stages of gradual refinement until the policy reaches an enforceable stage. One possible solution could be to push the need to know all about the resources and their corresponding actions as late in the policy engineering lifecycle as possible. This would not only assist policy authoring by abstracting away (frequently changing) details about the managed system, but can provide quality *feedback* as well, in case of policy conflicts. This can be achieved by having an up-to-date snapshot of the resources and

their corresponding actions in question into a structured, manageable and easy to query knowledge base (KB). This KB can be represented using ontological representations. This is because the use of ontologies can provide a centralized form of knowledge aggregation, when even complex relationships can be described and queried. Using OWL DL [7] for instance, the "equivalent" and "disjoint" relationships between resources and actions can be described. In our example, devices having different operating systems could be instances of subclasses describing their respective operating system, which in turn belong to a common super-class of 'operating systems'. However, in terms of performing software updates, as specified in our example, those subclasses need to be disjoint. This information can potentially be beneficial for policy authors in avoiding policy conflicts, e.g. by performing updates on the appropriate device only.

SWRL [8] uses the power of rules for rule enforcement and the power of OWL DL to form all rule predicates. One key difference between SWRL and the research presented in this paper is that we are investigating whether ontologies can provide benefits when used to model policy *targets* and their *actions,* but this is in a framework where attractive alternatives for modeling the specific interactions between policy *subjects* already exist. Role-based Access Control (RBAC) [9] already represents a widely accepted structure for modeling policy subjects. However, our previous work has identified that the need to capture the *flow of authority* is required [1]. Large organizations have decision-making groups, which are usually hierarchical in nature and often *delegate* the decision authority to other groups lower in the hierarchy. To model such concepts the use of the ontological class-subclass relationship is not adequate. For example, a company's Director has authority over the Project Lead, but the latter is not a subclass of the former. In reality however, decision-making groups are more fluid and overlapping than the classic hierarchical organizational models that underpin role-based schemes. For this purpose, this research will use the Community-based Policy Management (CBPM) [10] framework to model policy subjects, which has been shown to model the organizational aspects of policy subject models in a way that makes the resulting policy base more robust to organizational change [11]. In this paper we therefore aim to investigate how different combinations of DL and LP impact on the overall system expressiveness and on the cognitive load placed upon the requirements engineer, while aiming for a similar level of policy engineering robustness for target models, which CBPM delivers for subject models.

CBPM uses the notion of *Community* as the primary grouping abstraction with the aim of allowing groups within the organization itself to define communities to naturally reflect the changing nature of decision making (i.e. policy setting) authority. Communities towards the top of the hierarchy have the wider membership and more general function, while those toward the bottom have more narrow membership and more specific function. The hierarchy is designed to support agility and autonomy, allowing new sub-communities to be formed and encouraging the delegation of decision making authority as far down the hierarchy as possible. Using CBPM enables us to have a powerful *permissions scheme* in place. This scheme

grants or denies permission to perform actions on resources and it can also be used to define obligations.

III. DESCRIPTION LOGIC (DL) AND LOGIC PROGRAMMING (LP) INTEGRATION

Before we present in detail the integration details of OWL DL and CBPM, it is worth presenting an overview of the Description Logic and Logic Programming integration. The integration between ontologies and rules can be categorized into *homogeneous* and *hybrid* integration [2]. In the homogeneous approach, ontologies and rules are embedded into a new logical language L. The rule predicates that form the operational policies, use ontology concepts such as ontology properties or instances. SWRL falls into this category, as it is an extension of OWL DL axioms with rule axioms. SWRL rules can be used to define new classes and new properties of the ontology.

In the hybrid approach, there is a separation between the ontological and the rule predicates. The ontology remains unchanged and rules are built on top of ontologies.

A. CBPM and OWL DL integration

The type of integration between CBPM and OWL DL presented in this paper, is of the hybrid type, which means that there is no creation of a new logical language. The resulting framework is called *CBPM+DL*. CBPM+DL is formed as follows: a hybrid knowledge base K = (S,R) is a finite set S of DL axioms in the ontology language S and a finite set of hybrid rules R over R and S, including non-DL atoms. This definition follows the hybrid integration definition presented in [2].

Since the CBPM+DL method follows this hybrid approach, the semantics entailed in the hybrid language are derived from the semantics of its constituent components, namely CBPM and OWL DL. A hybrid rule r is obtained from the CBPM rule r′, by adding DL queries in the body. The latter are in fact additional constraints on the use of the rule r′, which have to be satisfied so that the rule r′ is fired.

B. CBPM+DL syntax

The syntax of CBPM+DL is formed by extending the existing CBPM syntax with ontological predicates.

Two of the most important type of rules that govern CBPM are *membership rules*, which define the rules for an entity to be a member of a particular Community and the *authorization rules*, which grant or deny permission to an entity to perform an action on a particular resource. Both membership and authorization rules have been extended with ontological predicates.

1) Community Membership Rule

The membership rule defines the restrictions that need to be satisfied, for a person P to be a member of Community C.

$Person(?P) \land property.?P \rightarrow memberOf(?P, Community)$

Property is of type ontological object property restriction. This means that user-specific properties can be created.

2) Authorization Rule

Authorization is about granting or denying access (negative authorization) to a subject to perform an action on a resource. In this research we use the notion of *resource authority,* which is introduced in CBPM. A resource authority specifies what action on which resource, access is sought for.

The resource authority is defined as follows:

Action(?a) ∧ Resource(?r) → ResourceAuthority(?ra)

The resource authority is used as a predicate in the authorization rule:

comm(?comm) ∧ isa(?x, ?y) → posAuth(?comm, ?ra)

The isa(?x, ?y) denotes the OWL is-a relationship, and in this case it means that x is a subclass of y. This rule can be read that as long as the "isa" relationship is satisfied, the community *comm*, is authorized to perform an action on a resource, as specified in the resource authority restriction.

3) Implies authority rule

The *implies authority* relationship describes the authority which is exercised by a Community, an Action, and a Resource on a Community′, an Action′ and a Resource′ respectively, which are located lower in the hierarchy of authority.

Community ImpliesAuthorityOver Community′
Action ImpliesAuthorityOver Action′
Resource ImpliesAuthorityOver Resource′
　　　′: lower in the hierarchical tree

For example, the Community called *Directors* ImpliesAuthorityOver the Community′ called *Projects Leads*. The Action *Read All* ImpliesAuthorityOver *Read File*. The Resource *Servers* ImpliesAuthorityOver *Database Servers*.

4) Resource authority Delegation rule

The delegation rule denotes that for a particular condition, the Resource Authority ?ra is delegated from community C to community C′, where Community C′ is lower in the hierarchy of authority than community C.

C. "Implies authority" vs "is-a" relationships

One of the main drives for this research in pursuing the integration of CBPM with OWL DL is the realization of the benefits that both technologies can offer in policy engineering. By using CBPM for modeling policy subjects, its ability of handling change at the organizational level can be utilized. This can be achieved through the use of its flexible *Community* grouping abstraction and some of its other native constructs, such as the "*implies authority*" relationship and the delegation of *resource authorities*. The OWL DL framework on the other hand is well known for its subsumption abilities (sub-class and sub-property relationships). For example, the ability to combine the "implies authority" with the subclass relationships has been proven extremely helpful in describing various scenarios. In the encoding of the TCD statutes case study that will be presented below, the importance in the co-existence of the two relationships will be highlighted.

IV. THE TCD STATUTES CASE STUDY

For the last 3 years the Trinity College Dublin (TCD) Statutes Review Working Party has been commissioned with the task of reviewing and rewriting the Trinity College Dublin Statutes. This well documented, real-world problem of statutes definition and interpretation in order to minimize the number of conflicts poses a big challenge. We decided to study the combination of CBPM and OWL DL techniques in order to tackle the encoding of existing statutes, the detection of conflicts when enforcing new and existing statutes.

Some of the challenges this effort has, is to capture the complexity of the organizational structure of the College, with the policies and obligations pertaining to each College body. For this purpose the use of the CBPM was chosen as a candidate to model policy subjects. Also, due to the reasoning power of OWL DL, static conflict analysis could be performed and this was one driving factor for using ontologies.

However, the main challenge that we faced was finding the right *balance* between the use of CBPM and OWL DL in terms of system expressivity without incurring a big penalty on the user *cognitive load*. Since it is envisaged that such a system would be used not only by experts in policy management, but also by domain experts (i.e. experts in law specification in this case), the creation of easy to use, custom tools that support these encodings is considered of great importance.

Five carefully selected participants took part in the user trial, each with a strong background on either policy-based management, ontologies, or logic programming. This is because the users would be asked to encode statutes using such techniques. The user trial involved a training session introducing all encoding methods, examples of already encoded statutes and finally the user statutes encoding session. The whole trial was recorded and in the end the users were asked to fill in a questionnaire.

All users started with CBPM, and if it was not possible to encode with CBPM, they moved to the next available encoding method, which is CBPM+DL. The same happened until the users traversed all remaining encoding methods, namely OWL DL, OWL DL + LP, and LP.

The OWL DL + LP method was in essence the use of the SWRL language. When selecting LP in this trial, the users had to use the syntax of a generic rule language such as Drools [12].

A. User trial outcome

Figure 1 provides a comparison between the different encoding methods in terms of the easiest method to use, which is an indirect indication of the user cognitive load experienced. On average, most users believe that CBPM+DL is the easiest encoding method to use, and is followed by CBPM, OWL DL, DL+LP and finally LP. CBPM+DL and especially CBPM score low in terms of ease of use for those users who have been exposed very little or not at all prior to the experiment training session.

Other findings (not shown in Figure 1) show that due to its inability of inferring conditionals OWL DL is thought of being

the method least suited for encoding statutes and organizational rules in general.
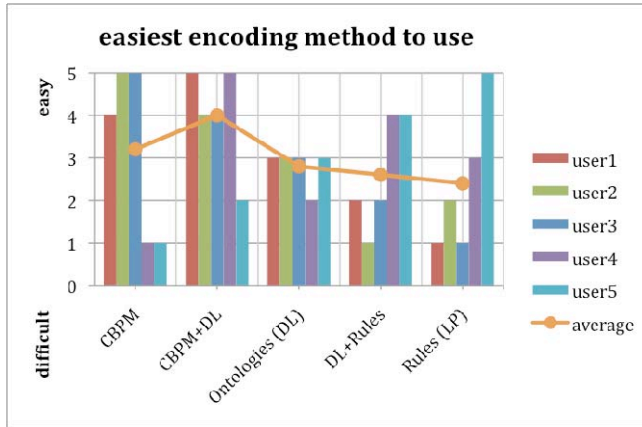

Figure 1: Easiest encoding method to use

In contrast, DL+LP is regarded as the most suited method for encoding statutes mainly because of its ability to easily describe conditionals. DL+LP is closely followed by CBPM+DL mainly due to the latter's ability to take advantage of the powerful CBPM constructs such as the "implies authority" relationship, which is coupled in this case with the ability to add user-defined properties, as well as "is-a" relationships.

CBPM is also featured prominently as suitable for encoding statutes, because of the reasons discussed so far, as well as due to its ability to express conditionals via restricting membership into Communities, albeit sometimes resulting in Communities that are against peoples' way of thinking.

V. DISCUSSION

The main purpose of this user trial was to test semantically different technologies such as Community-based Policy Management (CBPM), ontologies and rule-based systems in modeling real world scenarios. The modeling of policy subjects using CBPM and policy targets using OWL could be beneficial as shown in the security example in section II. What has become obvious from analyzing the user responses, is that there is no one-size-fits-all encoding method, which doesn't cause a rule clutter. Obviously, nearly every clause can be expressed in Logic Programs (LP) in the form of generic rules such as [12], but firstly, not all users are comfortable in using and interpreting LP rules and secondly the outcome being in the order of hundreds or thousands of rules, would be hard to maintain. Also, the benefit of having static rule conflict analysis that OWL DL provides, would be missing from an LP only case.

The introduction of CBPM+DL encoding method can be seen as a bridge between two different worlds. Using CBPM+DL in the trials, the users were able to create custom properties and intuitive encodings in the cases where CBPM produced counter-intuitive ones.

VI. CONCLUSIONS

This research aims to assist policy experts and domain experts in the task of IT policy engineering, and especially in policy specification and in policy conflict resolution when modeling real world scenarios, with often frequent organizational and IT resource change. The work shown in this paper focuses on finding the right balance between semantically different modeling techniques both in terms of expressivity, as well as in terms of the cognitive load experienced by the policy authors. Such an evaluation can help determine shortfalls in the design of the software system or of the policy model used. This work proposes a methodology based on the hybrid integration of Description logic (DL) and Logic Programming (LP), which utilizes the Community-based Policy Management framework for encoding policy subjects and OWL DL for modeling policy targets. In essence, this method extends CBPM with OWL DL predicates and it is called CBPM+DL.

Related work can be found in [13], where the authors present a semi-automatic way of encoding regulatory policies based on pre-defined templates.

REFERENCES

[1]   K. Feeney, C. Tsarouchis, and D. Lewis, "Policies as Signals in Collaborative Policy Engineering," presented at the Policy-based Autonomic Computing Workshop, PBAC, 2007.

[2]   G. Antoniou, et al., "Combining Rules and Ontologies. A survey," 2005.

[3]   F. Baader, et al., The Description Logic Handbook, 2002.

[4]   C. Baral and M. Gelfond, " Logic programming and knowledge representation," Journal of Logic Programming, vol. 19/20, pp. 73-148, 1994.

[5]   D. Batra, "Cognitive complexity in data modeling: causes and recommendations," Requirements Engineering, vol. vol 12, pp. 231-244, 2007.

[6]   B. Tsoumas and D. Gritzalis, "Towards an Ontology-based Security Management," presented at the Proceedings of the 20th International Conference on Advanced Information Networking and Applications, 2006.

[7]   OWL-overview. http://www.w3.org/TR/owl-features/.

[8]   SWRL. http://www.w3.org/Submission/SWRL/.

[9]   D. F. Ferraiolo and D. R. Kuhn, "Role Based Access Control," in 15th National Computer Security Conference, 1992, pp. 554-563.

[10]   K. Feeney, et al., "Relationship-Driven Policy Engineering for Autonomic Organisations," in 6th IEEE International Workshop on Policies for Distributed Systems (POLICY 05), 2005, pp. pp89 - 98.

[11]   R. Brennan, et al., "Policy-based integration of multiprovider digital home services," Netwrk. Mag. of Global Internetwkg., vol. 23, pp. 50-56, 2009.

[12]   Drools. http://labs.jboss.com/drools/.

[13]   G. Koliadis, et al., "Analyst-Mediated Contextualization of Regulatory Policies," presented at the Proceedings of the 2010 IEEE International Conference on Services Computing, 2010.