Policy-Based Governance of Complex Distributed Systems: What Past Trends Can Teach Us about Future Requirements

Jeffrey M. Bradshaw, AndrzejUszok
Florida Institute for Human and Machine Cognition (IHMC)
40 S. Alcaniz Street, Pensacola, FL 32502 – USA
{jbradshaw, auszok}@ihmc.us

Rebecca Montanari
Department of Computer Science and Engineering (DISI)
University of Bologna
Viale Risorgimento, 2 – 40136 Bologna – Italy
rebecca.montanari@unibo.it

1. Introduction

Policy-based management has been the subject of extensive research in recent decades [Robinson, 1988], [Sloman, 1989], [Chada et al., 2002], [Calo, 2003], [Boutaba, 2007]. The idea of adopting machine-enforceable policies as means of specifying and governing the behavior of distributed systems emerged as a result of increasing system and network management complexity [Moffet, 1993]. In order to cater to changing requirements, large distributed systems must be capable of changing and adapting their behavior while they are still running.

Policies are a means to dynamically regulate the behavior of system components without changing code and without requiring the consent or cooperation of the components being governed [Bradshaw, *et al.*, 2005], [Chada, 2002]. By changing policies, a system can be continuously adjusted to accommodate variations in externally imposed constraints and environmental conditions.

Policy research was originally focused on the problems of large-scale enterprise-wide or Internet-wide systems [Boutaba, 2007]. In these applications, policies have been exploited either to automate access control or to handle network administration tasks, such as configuration, security, recovery, or quality-of-service (QoS). Considerable effort has been applied to develop expressive representations and sophisticated management systems for specifying, managing, analyzing, and enforcing policies.

Since these early years, the scope of policy management has gone far beyond traditional distributed system and network concerns. New application fields for policy management include, among others, multi-agent systems, pervasive and mobile devices, and autonomic computing systems [Bradshaw, 2004, 2012], [Bunch, 2012], [Montanari, 2004], [Keeney, 2003], [Lupu, 2007] , [Xu, 2011]. The novel requirements of these additional application areas, in addition to problems that have arisen in large-scale deployments within more traditional applications, have posed a never-ending stream of challenges to policy researchers.

In this paper, we look at selected trends in research and development of policy management systems for complex distributed systems. We believe that a careful glance in the rear-view mirror will help researchers anticipate some of the crucial policy management requirements of the future—a future that portends ever-increasing scale, heterogeneity, decentralization, and complexity.

Rather than organizing this paper by application area, we have approached the subject topically, with a focus on a selection of trends that highlight general common considerations *across* applications that will be driving the direction of policy research and development into the future (Section 2 and Section 3). Section 4 will discuss additional implications of these trends, and will outline important challenges yet to be seriously addressed.

2. Progress and Prospects in Policy Specification Approaches

In this section, we outline the most adopted policy representation approaches in widely-known policy languages. In particular, we reflect on the research results and application demands that have motivated progress for that area. Then, we summarize our views on the most promising directions and needs.

2.1 Historical Overview

In network management, policies are applied to automate network administration tasks, such as security, configuration, recovery, or quality of service with the promise of reduced maintenance costs, improved flexibility, verifiability and runtime adaptability. The recognition of policy as a management approach can be traced back to the research of [Robinson, 1988], [Moffet, 1993], where management policies were described as sets of rules for achieving the scalable management of a distributed computing system and for achieving objectives of optimizing resource usage, cost, revenue, and performance [Boutaba, 2007]. Policy-based systems for system and network management typically distinguish two different kinds of policies [Chada, *et al*., 2002]. *Authorization policies* describe the actions that are allowed (positive authorization) or forbidden (negative authorization) to actors in a specified context, and are typically used for security purposes. *Obligation policies* define either the actions subjects must perform when certain conditions are triggered (positive obligation) or else exceptions to such requirements for subjects in a specific context (negative obligation or waiver). Obligation policies are typically exploited for configuration and QoS management.

There are a number of approaches to the definition of policies, and accompanying policy languages, which represent a number of different levels of policy expressiveness and policy enactment semantics. Multiple approaches for policy representation have been proposed, ranging from logic-based languages, to special-purpose policy languages that can be processed by machines, to generic rule-based (if-then-else) formats [Barnes, 1999], [Hoagland, 2000], [Damianou, 2000], to more recent ontology-based approaches [Bradshaw, 2003a, 2013], [Uszok, 2003, 2008, 2011], [Kagal, 2003], [Tonti, 2003]. Policy-based management suffers from fragmentation of approach, partly due to differences in semantics between access control policy languages and resource management policy languages. As a consequence there is no commonly accepted policy language and no common approach to the engineering of policy based systems, with many languages being proprietary in nature or tied to particular system management products or application scenarios.

Different schema can be used to describe proposed state-of-the-art policy languages. In the following we focus on the description of most widely adopted and cited policy languages, by analysing them from two different viewpoints, the policy representation model and the scope of the applicability.

2.2 Logic-Based Languages

Logic-based languages are attractive for the specification of security policy because they have a well-understood formalism that is amenable to machine inference [Damianou, 2000]. Examples of first-order logic based security policy languages include the logical notation introduced in [Chen, 1995]; the Role Definition Language (RDL) presented in [Hayton, 1998] and RSL99 [Ahn, 1999], and the authorization specification language (ASL) [Jajodia, 1997]. There are also proposals that exploit deontic logic for security policy representation. One early example had the aim of developing confidentiality policies that incorporated conditional norms [Glasgow, 1992]. In [Cholvy, 1997], deontic logic is used to represent security policies with the aim of detecting conflicts. In [Ortalo, 1998] deontic logic is exploited to express security policies in information systems. Despite the advantages of logic-based languages in terms of expressiveness and analyzability, they assume a strong mathematical background. This can make them difficult to use and understand. Moreover, they are not always directly translatable into efficient implementations.

2.3 Rule-Based Approaches

Ponder is the broadest, most influential, and most widely-deployed rule-based policy language [Damianou, 2001], [Sloman, 2002]. Ponder views policies as rules that define choices in system behavior that will reflect the objectives of the system managers. At a more formal level, a Ponder policy defines a relationship between objects (Subjects and Targets) of a managed system. Ponder2, the latest incarnation of this language, is a declarative, object-oriented language that supports the specification of several types of management policies for distributed object systems and provides structuring techniques for policies to cater for the complexity of policy administration in large enterprise information systems [Twidle, 2009].

A set of tools and services were developed for the specification, analysis and enforcement of Ponder policies. Thus, the name Ponder became associated not only with the language but with the entire toolkit. Ponder2 combines a distributed object management system with a Domain Service, Obligation Policy Interpreter, Command Interpreter and Authorization Enforcement capable of specifying and enforcing both authorization and obligation policies [Twidle, 2009]. The Domain Service provides an hierarchical structure for managing objects. The Obligation Policy Interpreter handles Event, Condition, Action rules (ECA). The Command Interpreter accepts a set of commands, compiled from a high-level language called PonderTalk, via a number of communications interfaces that may perform invocations on a ManagedObject registered in the Domain Service. The Authorization Enforcement caters for both positive and negative authorization policies, provides the ability to specify fine-grained authorizations for every object, and implements domain nesting algorithms for conflict resolution.

Other rule-based policy systems adopt an Event-Condition-Action rule paradigm. A popular example is the policy description language from Bell Labs [Lobo,1999] in which a policy is a function that maps a series of events into a set of actions. The language can be described as a real-time specialized production rule system to define policies. It consists of a policy rule corresponding to an obligation policy and of a rule for triggering other events. Policy rules map series of events into sets of actions. The language has clearly defined semantics and has been deliberately limited to a policy specification that is succinct and can be efficiently implemented.

A more recent example is IETF's Common Policy Language (CPL), a standard for representation of both authorizations (with related obligations) as Condition-Action rules, with an encoding in XML [Verma, 2000]. We note that recent work on CPL seems to have been focused exclusively on geographic location privacy for the Web. RuleML is a generic rule language implemented with the structure of Horn clauses (Head <- Body), but is evolving toward event-condition-action rule formats. It is sponsored by an international non-profit organization. DMTF's CIM-SPL (Simplified Policy Language) is a very simple rule language to represent obligation policies in "if-then" form. There is no support for authorization policies [Verma, 2000]. The TMForum Information Framework (SID) allows the semantic description of a complex set of interrelated instances, supporting Event-Condition-Action rules that could be used to support obligation policies. The Object Management Group's Semantics of Business Vocabulary and Rules (SBVR) provides a semantics for business rules. It specifies an XMI encoding of UML instances, with some semantic technology support via mapping to ISO Common Logic. Version 1.0 was released in January 2008, with no further updates and no known implementation [OMG, 2008].

2.4 General-Purpose versus Limited-Purpose Policy Languages

The languages described in Section 2.3 represent general-purpose policy approaches. Ponder2 can be considered the most significant example of general-purpose policy language since it is designed as an

extensible framework that can be used at widely different levels of scale — from small, embedded devices to complex services and Virtual Organizations.

In addition to general-purpose policy languages, there are a variety of limited-purpose languages that have been developed. Here, our focus will not be on product-specific policy languages (e.g., the Impact Policy Language that is part of IBM Tivoli Business Service Manager) or on simple approaches based on table look-up (e.g., firewall or router policies), but rather will highlight two examples of more broadly-conceived languages that were designed to work within a given family of operating environments (e.g., Web Services) or to serve specific requirements of particular kinds of applications (e.g., access control, digital rights management).

One of the most important of these languages is the Web Services Policy Language (WS-Security Policy) specification, an important standard that was developed by a wide consortium from the computing and security industry and has been adopted by the OASIS group [WSPL, 2006]. They define a set of XML-based security policy assertions (e.g., protection assertions requiring signing and encryption, token assertions specifying allowed token formats, and supporting token assertions that add functions such as user sign-on with a username token) that apply to SOAP Message Security, WS-Trust, and WS-Secure Conversation. Policies either can be used at development time to generate code with specified properties, or they can be used during runtime negotiation of security requirements for Web Service communication. Policies may be attached to WSDL elements. The use of XML for encoding allows great flexibility and ongoing evolution with respect to details in message-based communication with services.

One of the most widely-used policy languages for access control is XACML (eXtensibleACcessControl Markup Language), the subject of a very active OASIS standards group [XACML, 2013]. A number of implementations for XACML are documented on the Web site, with varying licensing terms, including some that are available for public download. Although XACML was originally conceived as an approach for access control (expressed as Condition-Effect Permit or Deny rules), it is now expanding its reach to additional problems. In some cases, obligations related to access control rules can also be specified, but the range of semantics is limited. Like WS-Security Policy, XACML uses an XML encoding for policies.

A variety of rights expression languages are used to provide machine interpretable encodings for digital rights management. The expressions themselves are typically embedded within metadata associated with the media. For example, ODRL (Open Digital Rights Language) is an open standard for an XML-based rights expression language (http://www.w3.org/community/odrl/). ODRL is implemented in three serializations: XML, RDF/OWL Ontology, and JSON.

2.5 Recent Directions in Policy Languages: OWL-Based Approaches

One of the major advantages of using XML as a policy representation is its straightforward extensibility. A problem with using XML (and many other policy representations) is that its semantics are mostly implicit — meaning is conveyed based on a shared understanding derived from human consensus.Implicit semantics based on convention have potential for ambiguity, often suffer fragmentation into incompatible representations, and require extra effort that could be obviated by a richer representation. For this and other reasons, semantic technologies are replacing XML as a representation of choice in many demanding application domains [Garcia, 2005]. Semantic technologies such as the W3C standards RDF and OWL are built using XML as a foundation, thus sharing its extensibility. However, in contrast to XML, they are able to represent and reason over rich information in great efficiency.

The Web Ontology Language (OWL) is a family of knowledge representation languages based on Description Logic (DL) with a representation in RDF [OWL, 2004]. OWL supports the specification and use of ontologies that consist of terms representing individuals, classes of individuals, properties,

and axioms that assert constraints over them. The axioms can be realized as simple assertions (e.g., Woman is a subclass of Person, hasMother is a property from Person that is inherited by Woman, Woman and Man are disjoint classes) and also as simple rules. OWL contains the necessary constructors for formal description of most policies and information management definitions [OWL, 2004], [Lopez de Vergara, 2004].

There are several advantages to OWL for policy representation. Ontologies simplify the task of governing the behavior of complex environments. The possibility of representing entities and changes in the behavior at multiple levels of abstraction improves the global *expressiveness*. The use of ontologies permits the policy framework to be easily extended, by simply adding new concepts to the ontology. In fact, any policy element (e.g., system components, actions, and context) can be described by appropriate concepts and relationships at the desired level of abstraction. In addition, because the semantics of such representations typically are a superset of the semantics of specialized "niche" policy languages, it is possible to convert ontology-based representations into the more specific languages. In traditional languages this task is usually much trickier. For example, in Ponder the specification of a communication policy example requires the non-trivial effort to extend the policy language or to convert the policy into a resource control policy. In addition, the possibility of modeling policies at a high-level of abstraction allows users to focus their attention more on high-level management requirements than on implementation details.

An ontology-based description of the policy enables the system to use concepts to describe the environments and the entities being controlled, thus simplifying their description and improving the *analyzability* of the system. As a result, policy frameworks can take advantage of powerful features such as policy conflict detection and harmonization [Guerrero, 2006]. In addition, ontology-based approaches simplify access to policy information, with the possibility of dynamically calculating relations between policies and the environment, entities, or other policies, based on ontology relations rather than fixing them in advance. Like databases, it is possible to access the information provided by querying the ontology according to the ontology schema. This is an advantage in comparison to traditional languages that provide only pre-defined queries to access information and static representations of policy. How to design the ontology is an application-dependent problem. As in database design, ontologies should be designed and extended consistent with the application context and optimized for the most common queries. Finally, ontologies can also simplify the sharing of policy knowledge among different organizations and applications, thus increasing the possibility for entities to negotiate policies and to agree on a common set of policies. Table 1 compares ontology-based approaches to policy to the Ponder, as a mature representative of the non-ontology-based general-purpose policy language approach.

	Semantic web languages for policy specification	Ponder ** ** used as example of non-semantic web language
Expressiveness	Capable of representing concepts and behavior of any complex environment	Capable of controlling specific sorts of behavior within object-oriented systems
	Multiple levels of abstraction	Low level of abstraction: object level
	Easy to extend policy ontology at runtime with new concepts	Extensibility supported by object-oriented inheritance at compile-time
Analyzability	Ontology representation simplifies and directly supports policy reasoning, conflict detection and harmonization	Conflict detection requires transforming policy specification into an event calculus representation
	Simplified access to policy information by querying the ontology	Access to single policy object by API – Access to policy repository to be designed
Ease-of-use	Need of specialized tools to assist unskilled users with policy specification and interpretation	Language specifically designed for simple policy specification and direct readability
Enforceability	High-level specification requires skilled programmers or sophisticated policy automation mechanisms for enforcement	Detailed specifications can be directly mapped into policy enforcement mechanisms
	Policy sharing among heterogeneous systems requires an agreement on a common ontology	Policy sharing among heterogeneous systems requires agreement on interfaces

Table 1. Comparison between OWL-based and Ponder approaches (Tonti, 2003)

All this being said, the adoption of ontologies for policy specification requires addressing some technical difficulties. OWL representations present a complex syntax, long declarative descriptions, hyperlinks, and references to external resources that can make it difficult to read (e.g., compare the readability of a Ponder policy with an OWL policy). To improve the *ease of use* of these languages, graphical interfaces or other tools for policy specification are necessary. In addition, *enforceability* is a critical aspect for the OWL approach. Ontology-based policy specification can be difficult to implement in comparison to other policy specification such as Ponder, because of the high-level specification of ontology-based policies can be far removed from the concrete implementation of the policy enforcement on the systems. Usually the gap between the specification and the implementation of policies cannot be completely overcome in an automated manner, but has to be resolved to a greater or lesser degree by human programmers, consistent with the capabilities and features of each platform. Enforcement code generation facilities and libraries of enforcement mechanisms adapted to specific platforms are among the most important features for OWL-based policy management frameworks to provide to enable their widespread implementation.

In the following we describe the key characteristics of some widely-known semantic-based policy representations KAoS, Rei, and AIR so that to evaluate the advantages and disadvantages of adopting semantic technologies for policy representation in practice.

2.5.1 KAoS. IHMC's KAoS policy services framework is a mature general-purpose policy management system that relies on OWL in the specification, analysis, and enforcement of policy constraints across a wide variety of distributed computing platforms [Bradshaw, 2003a, 2013], [Uszok, 2003, 2008, 2011]. KAoS supports both authorization and obligation policies and supports specialized policies and mechanisms of other kinds of policies (e.g., delegation management, policies about space and time). KAoS enables the specification, management, analysis, and enforcement of policies. It provides the KAoS Policy Administration Tool (KPAT) as a graphical interface to assist users in policy specification, revision, and application. In addition, KPAT can be used to browse and load ontologies and to analyze and deconflict newly defined policies. It supports static (description-logic-based) and

dynamic conflict resolution (collective obligation and planning/resource allocation mechanisms), and the enforcement of policies through a separate software element called the Guard. The breadth of its semantics have allowed its use across multiple application domains and operating environments, including intermittent tactical networks and standalone sensors.

KAoS is implemented in Java. It has well-defined interfaces for programmatic access of functionality and the ability to import and export OWL ontologies. Over the last decade, several government agencies and private organizations have sponsored research and development efforts to mature KAoS. KAoS has been enhanced for scalability, more powerful and flexible reasoning, and for use within distributed enterprises. KAoS has been integrated into IHMC's Luna agent framework [Bunch, 2012], as well as several other agent platforms and traditional service-oriented architectures. IHMC and its collaborators have undertaken to translate ontologies to third-party approaches in previous efforts and are currently working to extend these efforts into new arenas (e.g., translation from KAoS ontologies to XACML, translation from natural language documents to KAoS ontologies).

2.5.2 Rei. Rei is a policy framework that integrates support for policy specification, analysis and reasoning in pervasive computing applications [Kagal, 2003]. Rei has been used in conjunction with the Vigil security framework, Fujitsu's Task Computing project, and with the Groove workspace [http://www.groove.net] within the DARPA Genoa II program [http://www.darpa.mil/iao/GenoaII.htm]. The Rei deontic concept-based policy language allows users to express and represent the concepts of rights, prohibitions, obligations, and dispensations. In addition, Rei permits users to specify policies that are defined as rules associating an entity of a managed domain with its set of rights, prohibitions, obligations, and dispensations. Rei relies on an application-independent ontology to represent the concepts of rights, prohibitions, obligations, dispensations, and policy rules. This allows different elements of a pervasive environment to understand and interpret Rei policies in the correct way. In particular, Rei adopts OWL-Lite to specify policies and can reason over any domain knowledge expressed in either RDF or OWL A policy basically consists of a list of rules expressed as OWL properties of the policy and a context represented in terms of ontologies that is used to restrict the policy's applicability. Though represented in OWL-Lite, Rei still allows the definition of variables that are used as placeholders as in Prolog. In this way, Rei overcomes one of the major limitations of the OWL language, and more generally of description logics. i.e., the inability to define variables. On the other hand, the choice of expressing Rei rules similarly to declarative logic programs prevents it from exploiting the full potential of the OWL language. In particular, the Rei engine is able to reason about domain-specific knowledge, but not about policy specification. There seems to have been no further development activity on Rei since 2005.

2.5.3 AIR. AIR (Accountability in RDF) is a Semantic Web-based rule language that supports reasoning on the open Web (particularly trust and privacy issues) while focusing on generating and tracking explanations for its inferences and actions, and conforming to Linked Data principles (see [Air, 2007]). It relies on Web standards, though its implementation is proprietary. AIR uses Turtle (Terse RDF Triple Language) and N3 (Notation 3) rule encodings. AIR is not a general-purpose policy language, but focuses specifically on trust and privacy issues.

3 Selection of Research Issues Relating to Policies and Adaptation

3.1 Policy Enforcement and Adaptation

Policies tend to be seen as prescriptive and externally-imposed rules whose enforcement ensures a predictable system behavior. Policy-based enforcement approaches exhibit typical features:

1. work involuntarily with respect to the system components that are governed by policies, that is, without requiring the system components to consent or even be aware of the policies being enforced, thus aiming to guarantee that the system complies with policy;
2. wherever possible, are enforced preemptively, preventing in advance buggy, poorly designed, unsophisticated, or malicious system components from doing harm;

These characteristics are well suited for traditional distributed system/network management where it is crucial to ensure that a system behaves within well defined boundaries. Considering that in recent years the scope of policy managementhas enlarged, some adjustments to policy enforcement approaches are required to address the specific features of the novel application fields.

3.1.1 Self-Regulation of Compliance

In traditional distributed systems guarantees of policy conformance are typically required. These guarantees are assured through the use of an independent policy enforcement component, independent of the reasoning mechanisms in the software components being regulated (i.e., the subjects of policy).

Some research efforts, especially those who are studying the development of norms and the influences that determine their degree of adoption in artificial social communities, have looked at issues of self-regulation. In such efforts, the subject of policy rely to a greater or lesser degree on their own reasoning to determine whether or not to adopt a given policy or norm, and may engage in negotiation with other subjects about norm compliance (e.g., [Grossi, 2006], [Sen, 2007]). The need for this kind of self-regulation arises especially in multi-agent systems where the design and development of policy-based frameworks have to deal with a crucial agent dimension, i.e., agent autonomy, intended as the capability of an agent to take care of itself and the quality of freedom from outside control [Bradshaw, 2003b].

Research on adjustable autonomy in agent systems proposes a hybrid between models of self-regulation and external enforcement. The coupling of autonomy with policy mechanisms allows agent designers to achieve adjustable autonomy with the primary purpose to maintain the system being governed at a sweet spot between convenience (i.e., being able to delegate every bit of an agent's work to the system) and comfort (i.e., the desire to not delegate to the system what it cannot be trusted to perform adequately).Adjustable autonomy gives the agent maximum freedom for local adaptation to unforeseen problems and opportunities while assuring humans that agent behavior will be kept within desired bounds.

Adjustable autonomy requires general engines for reasoning about how to adapt policies and resources to improve system performance and effectiveness for end-users (on whose behalf agents act in the system), while respecting absolute security constraints set by administrators. Along this direction some initial work has been already performed focusing specifically on trading off operational necessity and security risk. In [Bradshaw, 2005] formalisms and mechanisms are presented that have been developed for adjustment of policies and resources that will facilitate effective coordination across distributed systems. Kaa (KAoS adjustable autonomy) is a component of KAoS that permits it to perform such adjustments or semi-automatically. Assistance from Kaa in making autonomy adjustments might typically be required when it is anticipated that the performance of the current configuration has led to or is likely to lead to failure or unacceptable performance, or when there is no set of competent and authorized humans available to make such adjustments themselves. Ultimately, the value of performing an adjustment in a given context is a matter of expected utility: the utility of making the change vs. the utility of the status quo. The current implementation of Kaa uses influence-diagram-based decision-theoretic algorithms to determine what if any changes should be made in agent

autonomy. However, Kaa is designed to allow other kinds of decision-making components to be plugged-in if an alternative approach is preferable. When invoked, Kaa first compares the utility of various adjustment options (e.g., increases or decreases in permissions and obligations, acquisition of capabilities, proactive changes to the situation to allow new possibilities), and then-if a change in the status quo is warranted-takes action to implement the recommended alternative.

3.1.2 Context-Based Policy Adaptation

In several application fields, e.g., multi-agent systems, mobile and autonomic systems, another crucial requirement is the possibility to dynamically change policies to adapt system behaviour to unpredictable contexts of operation. Changing policies means, for example, to prolong the validity of acquired permissions even in presence of changes in the conditions that have made the policy applicable or to find alternative permitted/obliged actions in the case the permitted/obliged actions as determined by the current state of the world cannot be performed because of constraints inherent in the current situation. Addressing policy adaptation requires to identify both appropriate policy representation and enforcement models.

It has been recently recognized that a progress in developing adaptive policy systems is represented by the adoption of a policy model that takes context into account. Whereas traditional systems rely on a relatively static characterization of the operating conditions where changes in the set of both clients (users/devices) and accessible resources are relatively small, rare, or predictable, new application scenarios are characterized by frequent changes in physical user location, in accessible resources, and in the visibility and availability of collaborating entities. The conditions defined at design time to control and govern resource operation and sharing can be unpredictably different from the ones that actually hold at execution time when entities attempts to access some resources. Novel context-aware policy models/systems should be conceived to take into account the high unpredictability, heterogeneity, and dynamicity of the new application fields. Whereas in traditional policy models context is an optional element of policy definition that is simply used to restrict the applicability scope of policies, in context-centric solutions context is the first-class principle that should explicitly guide both policy specification and enforcement process. It should not be possible to define a policy without the explicit specification of the context that makes that policy valid. Context-aware policy models enable policy adaptation where with the term “adaptation” we mean the ability of the policy management system to adjust context and policy specifications in order to enable policy enforcement in different, possibly unforeseen situations. For example, policy adaptation may allow to identify an alternative context where permitted/obliged actions can be performed. Context adaptation can be useful to handle the case of dynamic policy conflicts, such as when an entity obliged to perform an action cannot perform it because it is not allowed to.

Considering context as a first-class design principle is a very recent research direction with some context-driven policy model proposals, mainly in the field of access control. The importance of taking context into account for securing pervasive applications is particularly evident in [Covington, 2001] that allows policy designers to represent contexts through a new type of role called environment role. Environment roles capture relevant environmental conditions that are used for restricting and regulating user privileges. Permissions are assigned both to roles (both traditional and environmental ones) and role activation/deactivation mechanisms regulate the access to resources. Environmental roles are similar to our contexts in that they act as intermediaries between users and permissions.

Proteus is a context-aware policy model that is centered around the concept of context and that exploits semantic-based representation of context and policies [Toninelli, 2007]. Proteus contexts act as intermediaries between entities and the set of operations that they have to and/or can perform on resources. Proteus policies define for each context how to operate on resources. In particular, policies can be viewed as one-to-one associations between contexts and allowed/obliged actions. Entities

should and/or can perform only those actions that are associated with the contexts currently in effect (active context), i.e., the contexts whose defining conditions match the operating conditions of the requesting entity and of the environment as measured by specific sensors embedded in the system. [Ko, 2006] proposes an approach that allows to overcome the semantic gap between contexts specified in the policy at design time and contexts collected from dynamic context sources in pervasive environments: an access request is allowed if the query context is semantically equivalent to the context specified in the policy rule. The policy model in [Ko, 2006] also exploits semantic technologies. In particular, contexts and policies are defined by adopting an OWL-based representation, and OWL inference rules are exploited to derive relationships among contexts.

Context-based policy models require, however, to take into account the quality of context information used to drive policy decisions (QoC). Quality of context has in fact a profound impact on the correct behavior of any context-aware policy framework. Depending, for instance, on the quality of used context data, granting access to a resource might be associated to a variable risk level: the less reliable context information is (i.e. the lower its quality), the higher risk is associated to any access action allowed based on that context information. Using context information with insufficient quality might increase he risk of incorrect access control decisions, thus leading to dangerous security breaches in resource sharing. The importance of considering QoC in designing and managing context-aware systems has recently started to be recognized, with few proposed solutions [Buchholz, 2003], [van Sinderen, M., 2006], [Toninelli, 2007].

3.2 Adaptation Considerations at Runtime

The work on policies for system/network management so far has mainly focused on large, high bandwidth, fixed networks, e.g., enterprise networks, content provider networks, Internet service provider (ISP) networks, etc. Therefore, policy deployment is often based on centralized provisioning and decision-making. The ongoing standardization efforts toward common policy information models and frameworks witness the adoption of centralized architectures for policy enforcement [Verma, 2000]. IETF and DMTF have jointly produced a set of standards on policy information models and policy management architectures. The two main elements in their model of a policy management system are Policy Decision Point (PDP), a logical entity that makes policy decisions for itself or for other network elements that request such decisions; and Policy Enforcement Point (PEP), a logical entity that enforces policy decisions. PDP is likely to store its policies in a repository, such as a Lightweight Directory Access Protocol (LDAP) directory service. The basic interaction between the components begins with the PEP. The PEP will receive a notification or a message that requires a policy decision. Given such an event, the PEP then formulates a request for a policy decision and sends it to the PDP. The PDP returns the policy decision and the PEP then enforces the policy decision by appropriately accepting or denying the request.

The choice of the architectural model for a policy management system is one of the key factors in the success of such a system and impacts on the overall run-time system functioning efficiency. An architecture advantageous in managing a particular network environment (e.g., high bandwidth enterprise network) may not be an appropriate choice for managing another network (e.g., low bandwidth mobile wireless network). Hence, it is important to study the performance trade-offs involved and choose an architecture (or combination of architectures) that suits the requirements of the deployment scenarios. A plethora of policy management architectures have been proposed for as many environments as differentiated services, enterprise networks, utility computing, data centers, wireless networks, optical networks, and middleware systems. However, there is still no real large-scale deployment of policy management as of today.

A taxonomy of policy architectures have been proposed in [Phanse, 2006] based on various characteristics: a) the locus of control that represents one or more policy servers or policy decision

points (PDPs) in a network, capable of making policy-based decisions; b) the locus of information that refers to the location of the policy information storage module or repository used to store policies and accessed by a policy server to make decisions; c) the policy distribution model that defines the transfer of policy information between different points in the system; d) the tiers of control defined as the levels in a network where policy decisions are made. The work in [Phanse, 2006] distinguishes three main types of architectures: outsourced, provisioned and hybrid. In outsourced architecture all policy decisions are made at a single control tier, controlling the underlying nodes or PEPs. Unlike the outsourced approach, in the provisioned architecture at least two control tiers are involved, i.e., the locus of control and the locus of information are distributed across two or more tiers. The hybrid approach combines features of centralized and distributed architectures. From the analysis study conducted in [Phanse] on the various policy architecture types, it emerges that in order to extend the policy-based approach to newer application fields and networking environments, it is crucial to address the fundamental challenge of adapting the conceptually centralized idea of policy management to a decentralized paradigm applicable to the novel deployment environments. Especially in ad hoc networks there is the need to build a management framework that is as automated as possible, requiring minimal human intervention, and is intelligent, meaning it is able to learn about changes in networking conditions. This would lead us to a self-organizing or adaptive control structure that automatically reacts to network dynamics.

When deploying a policy-management system there are additional runtime considerations to address especially when self-regulating and/or context-aware policy enforcement approaches are adopted. For instance, in the case of self-regulating policy enforcement, evaluating options for reallocating tasks and change permissions/obligations among autonomous entities comes at a cost and may introduce performance overhead due to the need of entities to communicate, coordinate and reallocate responsibilities [Bradshaw, 2005]. When adopting context-aware policy approaches additional issues have to be addressed that may increase performance overhead. One significant performance penalty factor derives from the need to integrate context management services within policy frameworks. When, for instance, a resource access request is performed, the context associated to the request needs to be acquired and matched within the context data defining and activating the policy. In addition, context-aware policy models introduce additional complexity due to the need of evaluating the quality of context information associated to the policy request.

3.3 Adaptation As Part of Policy Conflict Resolution, Policy Refinement, and Polycentric Governance

Policy-based management still raises several research issues that have been partially solved. One concern when exploiting policies for ruling system/network behavior is the possibility of conflicts among policies, especially in situations requiring runtime resolution. After exploring current research in policy conflict resolution, we discuss adaptation through policy refinement and through polycentric governance.

3.3.1 Policy Conflict Resolution

There are a number of different conflicts that can arise from policies. Conflicts can arise in the set of policies. Conflicts can be modal conflicts, for instance where a positive and negative authorization apply to the same objects, or application specific conflicts related to the semantics of the resources and roles in the target and subject domains of policies, such as when two policies permit the same manager to sign checks and approve payments may conflict with an external principle of separation of duties [Lupu, 1999]. Conflicts may also arise during the refinement process between the high-level goals and the implementable policies. Some progress has been made in dealing with policy conflicts, eventhough significant challenges remain to be addressed, such as detecting conflicts when arbitrary conditions restricting the applicability of the policies. Chomicki and Lobo present in [Chomicki, 2003] a formal

logic-based framework for detecting and resolving action conflicts in ECA policies. Bandara presents in [Bandara, 2004] a tool for policy analysis. The tool supports querying a set of policies for validation and review. Validation queries are supported in order to determine the feasibility of a policy. Review queries are used in order to help the administrator analyze the managed system specification and extract specific types of information. In addition, Bandara suggests the use of abducting reasoning and the tool developed for event-calculus-based goal elaboration in order to query potential conflicts between policies.

Like many other systems, KAoS originally relied exclusively on numeric policy priority assignments by users to determine how policies should be ranked and deconflicted. A disadvantage of this approach has been that people may have difficulty assigning meaningful priorities and tracking how a given policy's priority relates to the priorities of other policies, especially when integrating large numbers of policies from different sources. For this reason, KAoS has been extended to use a logical precedence mechanism in addition to numeric priorities [Bradshaw, *et al.*, 2013]. This allows administrators to specify an almost-infinite variety of precedence relationships among policies, mirroring the kinds of rationale that people use when deciding which policy will trump another (e.g., policies defined by person A take precedence over anyone else's policies; policies of the domain administrator (a role) take precedence over user (another role) policies; more recent policies take precedence over older policies; policies about writing to a specific directory take precedence over policies about writing to the volume; negative authorizations take precedence over positive authorizations).

A final area of ongoing research in policy conflict resolution is for mechanisms to cover situations when specified policies may require more resources than are available in the environment and that need to have, where possible, a fair method of allocation under constraints specified as part of the policy (resource deconfliction). Preliminary work on this problem has been undertaken in the context of quality-of service policies for network operations [Loyall, 2011].

3.3.2 Adaptation Through Policy Refinement

Policy refinement is the process of deriving a more concrete specification from a higher-level objective. Although the goal of automating refinement of management and security policies from higher-level objectives remains a worthy long-term goal, it is currently not practical except in simple scenarios. A better near-term objective is for tools that provide partial automation of this process through assisting human managers to refine high-level abstract policies into more concrete ones. In [Javier, 2006] a methodological approach towards the policy refinement problem is provided. In particular, a generic procedure to define policy hierarchies is described, which is essential to achieving systematic policy refinement, as well as a policy refinement framework that formalizes the requirements to refine high-level guidelines into executable policies. Initial steps towards a framework for automated distributed policy refinement for both obligation and authorization policies are presented in [Craven, 2010] where the process of policy refinement is described as comprising three aspects: decomposition, operationalization and distribution. In policy decomposition, which is the focus of the work in [Craven, 2010], policies expressed at higher levels of abstraction are mapped into lower-level policies. The mapping is achieved using policy-independent refinement rules, defined within the scope of an application-specific system model. KAoS provides a "scenario" feature enabling the manual creation of high-level policies to guide the automatic generation of low-level policies. Further research is also needed on defining interfaces for the exchange of policies between the application and the hardware levels in order to effectively enforce policies defined at higher levels.

Typical approaches for addressing policy refinement in real-world systems, such as rule-based action policies, begin to suffer as systems become increasingly distributed, complex, and dynamic in nature. To support policy refinement at runtime, static mapping approaches for policy refinement will

need to be replaced by advanced dynamic methods, supported by a planner. Goal policies and real-time utility-based evaluation of policy effectiveness have recently started to be recognized as an attractive basis for representing and managing high-level objectives at runtime [Kephart, 2007]. Rather than specifying exactly what to do in a specific situation, goal policies specify either a single desired state or one or more criteria that characterize an entire set of desired states. The policy author specifies desired states as strategic goal policies (e.g., commander's intent) and specific tactical policies that best satisfy the needs of the current context would be automatically generated at runtime. The selection of tactics would be based on dynamic measures of utility.

3.3.3 Adaptation Through Polycentric Governance

Within the framework of resilient systems engineering, Branlat and Woods have discussed important patterns that lead to failure in complex systems [Woods and Branlat, 2008]. "Bottom-up" approaches to policy-refinement can be used to provide support for adaptive performance in the face of stressors and surprise through the principles of polycentric governance [Ostrom, 2008]. A related notion of organic resilience [Carvalho, 2010] was inspired by the concept of "organic computing" proposed in [Müller-Schloer, 2004]. Organic resilience relies heavily on biologically-inspired analogues and self-organizing strategies for the management and defense of distributed complex systems. The concept focuses on the design of emergent coordination mechanisms through local gradients and implicit signaling.

The use of collective obligations [van Diggelen, 2009a, b] is critical for practical applications of polycentric governance. Whereas an individual obligation is a policy constraint that describes what must be done by a particular individual, collective obligations are used to explicitly represent a given agent's responsibilities within a group to which it belongs, without specifying in advance who must do what. In other words, in a collective obligation, it is the group as a whole that becomes responsible, with individual members of the group sharing the obligation at an abstract level.

The execution and enforcement of collective obligations requires different mechanisms for different contexts. For some applications, a top-down policy refinement approach, implemented by a specialized planning system and spanning a group of agents, may be the best approach. However, in many cases a biologically-inspired "bottom-up" approach might prove more workable. Such an approach requires that the agents themselves, rather than some centralized capability, organize the work. This approach works best when agents themselves are in the best position to detect local triggers for collective obligations (e.g., potential threats or opportunities), to determine what support they can offer through their own resources and individual capabilities, and what information should be shared among peers and with agents elsewhere in the system. The self-organizing nature of the system enables the agents to revisit responsibilities and resource allocations themselves, as needed, on an ongoing basis.

Applied in a manner consistent with polycentric governance, we believe that policy-based collective obligations could provide the regulatory mechanisms to enable effective and coactive coordination algorithms for agents. Moreover, we envision the implementation of policy-learning mechanisms that could rapidly propagate lessons learned about productive and unproductive actions to whole classes of actors.

4. Looking Ahead

In the future, we expect the trend of policy to continue to evolve beyond single-purpose approaches that only deal with specific application niches. Instead, there is a desire to be able to formulate, manage, and enforce policies across an entire enterprise, elevating the specialized representations now used for access control, network security, device configuration, and so forth to a rich, general-purpose semantic specification [Uszok, *et al.*, 2011]. Only in this way can conflicts across diverse components and aspects of the system can be found and resolved, and can quality-of-service across the entire distributed

system be assured. Some of the current challenges and anticipated solutions by using an enterprise-wide ontology-based approach include the following:

- Sharing and integrating policies across multiple levels of an enterprise ->Ontologies can represent and harmonize policy from different perspectives and at multiple levels of abstraction by semantics, not by convention as must be done using XML
- Maintaining high-level strategic policy intent constant while adjusting tactical policies as the situation demands -> Ontology-based approaches for policy refinement, goal policies, and collective obligations
- Difficult to identify and resolve static and dynamic policy conflicts -> Efficient syntactic and semantic deconfliction and dynamic resource management algorithms
- Offline and online reasoning about policy, configuration management: if you make a change, the effects are not visible -> Use of rich semantics for real-time policy negotiation (e.g., risk-adaptive access control), what if analysis, dealing with rich context (e.g., history and state combinations)
- Growth in range of uses for policy leads to proliferation of different languages for different application niches -> Expressive ontology semantics can represent everything that is in the application-niche-specific languages and more; and can resolve inconsistencies within them and gaps between them
- Current implementations are not integrated, do not provide interfaces with other parts of an enterprise, nor ability to reason across domains (spectrum, cognitive radio configuration, network, authentication, logging, etc.) -> Can provide end-to-end system integration across multiple policy domains
- Need to specify policy by people who do not have specialized training -> Automatic tools can straightforwardly translate from natural language; other template-based tools allow point-and-click construction of ontology-based policy constraints
- Need to deal with legacy systems -> Automatic translation to application niche-specific languages
- Difficulty in managing large numbers of modular rule-based policies, implications of policies, gaps ->Graphical visualization tools for understanding policy at design time; ontologies present relationships among policy concepts as an integrated whole
- Standards driven by pragmatics of application needs not overarching principles -> collaborate with researchers, users, and vendors in developing a broad, principled approach

The foremost example of collaboration to develop a broad, principles approach is the NSA-sponsored Federal Digital Policy Management (DPM) initiative in the US. DPM has chosen an ontology-based approach to policy with sufficient semantics to subsume the more specific approaches to policy specification across all government agencies specifically for this reason [NSA, 2012]. DPM has adopted the KAoS core ontologies as the basis for future standards efforts [Digital Policy Management, 2012].

Though there may be reasons to translate the ontology-based policies into the specialized languages for the purpose of supporting legacy applications and specific enforcement components and devices (e.g., XACML, firewall policies), the focus is on specification and analysis across all application areas using a common ontology-based representation in OWL. Policy automation is a key concept in DPM. According to the organizers of the initiative: “Digital policies are an implementation where operating paradigms and access rules are created and maintained in executable formats that can be processed, downloaded to and enforced by IA devices. Digital Policy Management enables authorized operators to generate, adjudicate, validate, disseminate, and monitor policies.”

5 Conclusions

Even though each application area has stimulated the research on specific aspects of policy management and proposed specific solutions, all application areas share common considerations. As far as policy languages are concerned, policies can be specified in many different ways and multiple approaches have been proposed in different application domains, there are, however, some general requirements that any policy representation should satisfy regardless of its field of applicability: *expressiveness* to handle the wide range of policy requirements arising in the system being managed, *simplicity* to ease the policy definition tasks for administrators with different degrees of expertise, *enforceability* to ensure a mapping of policy specifications into implementable policies for various platforms, *scalabilty* to ensure adequate performance, and *analyzability* to allow reasoning about policies. The challenge is to achieve a suitable balance among the objectives of expressiveness, computational tractability, and ease of use for a given application. Other difficulties for the deployment of policy-based management on a large-scale derive from the technical challenges of providing efficient and adaptable policy run-time enforcement models and mechanisms.

References

[Ahn, 1999] Ahn, G.-J., et al., "The RSL99 Language for Role-based Separation of Duty Constraints", Proc. of the Fourth ACM Workshop on Role-Based Access Control, ACM Press, 1999.

[Air, 2007] Air Policy Language, http://dig.csail.mit.edu/TAMI/2007/AIR/).

[Bandara, 2004] Bandara, A. K., et al., "A goal-based approach to policy refinement", Proc. of the 5th IEEE International Workshop on Policies (Policy2004), IEEE Computer Society, 2004.

[Barnes, 1999] Barnes, J. F., et al., "CacheL: Language Support for Customizable Caching Policies", Proc. of 4th International Web Caching Workshop, San Diego, CA, 1999.

[Boutaba, 2007] Boutaba, R., "Policy-based Management: A Historical Perspective", Journal of Network System Management, Springer-Verlag, Vol. 15, No. 4, 2006.

[Bradshaw, 2003a] Bradshaw, J.M., et al., "Representation and reasoning for DAML-based policy and domain services in KAoS and Nomads", Proc. of the Autonomous Agents and Multi-Agent Systems Conference (AAMAS 2003), ACM Press, 2003.

[Bradshaw, 2003b] Bradshaw, J.M., et al., "Adjustable autonomy and human-agent teamwork in practice: An interim report on space applications", Agent Autonomy, Kluwer, 2003.

[Bradshaw, *et al.*, 2004] Bradshaw, J.M., Patrick Beautement, Maggie R. Breedy, Larry Bunch, Sergey V. Drakunov, Paul J. Feltovich, Robert R. Hoffman, Renia Jeffers, Matthew Johnson, Shriniwas Kulkarni, James Lott, Anil Raj, Niranjan Suri, and Andrzej Uszok. Making agents acceptable to people. In *Intelligent Technologies for Information Analysis: Advances in Agents, Data Mining, and Statistical Learning*, edited by Ning Zhong and Jiming Liu, 361-400. Berlin, Germany: Springer-Verlag, 2004.

[Bradshaw, 2005] Bradshaw, J.M., et al., "Kaa: policy-based explorations of a richer model for adjustable autonomy", Proc. of the 4th International Conference on Autonomous Agents and Multiagent Systems, (AAMAS'05), ACM Press, 2005.

[Bradshaw, 2012] Bradshaw, J.M., Marco Carvalho, Larry Bunch, Tom Eskridge, Paul Feltovich, Matt Johnson, and Dan Kidwell. Sol: An Agent-Based Framework for Cyber Situation Awareness. *Künstliche Intelligenz*: Volume 26, Issue 2 (2012), pp. 127-140.

[Bradshaw, *et al.*, 2013] Bradshaw, J.M. A. Uszok, M. Breedy, L. Bunch, T.C. Eskridge, P.J. Feltovich, M. Johnson, J. Lott, and M. Vignati. The KAoS Policy Services Framework. Poster and paper for presentation at the *Eighth Cyber Security and Information Intelligence Research Workshop (CSIIRW 2013)*. Oak Ridge, TN: Oak Ridge National Labs, January 2013.

[Buchholz, 2003] Buchholz, T., et al., "Quality of context: What it is and why we need it", Proc. of 10th International Workshop of the HP OpenView University Association (HPOVUA'03), 2003.
[Bunch, 2012] Bunch, L., J.M. Bradshaw, M. Carvalho, T. Eskridge, P. Feltovich, J. Lott and A. Uszok. Human-Agent Teamwork in Cyber Operations: Supporting Co-Evolution of Tasks and Artifacts with Luna. Invited Paper in Ingo J. Timm and Christian Guttmann (eds.), *Multiagent System Technologies: Proceedings of the Tenth German Conference on Multiagent System Technologies (MATES 2012)*, Trier, Germany, 10-12 October 2012. Berlin, Germany: Springer, LNAI 7598, pp. 53-67.
[Digital Policy Management, 2012] Digital policy management: Be part of the solution, not the problem. 2012. In *VanDyke Technology Group*. http://www.vdtg.com/blog/blog_post_policy_planning.html. (accessed May 29, 2013).
[Lobo, 1999] Lobo, J., et al., "A Policy Description Language", Proc. of the AAAI Conference, 1999.
[Calo, 2003] Calo, S. B., et al.., "Policy-based Management of Networks and Services", Journal of Network System Management, Springer-Verlag, Vol. 11, No. 3, 1994.
[Carvalho, 2010] Carvalho, M., T. Lamkin, C. Perez. Organic Resilience for Tactical Environments. In *5th International ICST Confernece on Bio-Inspired Models of Network, Information, and Computing Systems (Bionetics),* Boston, MA, December 2010.
[Chen, 1995] Chen, F., et al., "Constraints for Role-Based Access Control", Proc. of the First ACM/NIST Role Based Access Control Workshop, USA, ACM Press, 1995.
[Chada et al., 2002] Chada, R., et al., "Policy-based Networking", Special Issue, IEEE Network, Vol. 16, No. 2, 2002.
[Cholvy, 1997] Cholvy, L., et al., "Analyzing Consistency of Security Policies", Proc. of IEEE Symposium on Security and Privacy, IEEE Computer Society, 1997.
[Chomicki, 2003] Chomicki, J., et al., "Conflict Resolution Using Logic Programming", IEEE Transactions Knowledge Data Engineering, Vol. 15, No. 1, 2003.
[Covington, 2001] Covington, M.J., et al.: "Securing Context-Aware Applications Using Environmental Roles", Proc. of the 6th ACM Symposium on Access Control Models and Technologies (SACMAT 2001), ACM, 2001.
[Craven, 2010] Craven, R., et al., "Decomposition techniques for policy refinement", Proc. of the 6th International Conference on Network and Service Management, IEEE Computer Society, 2010.
[Damianou, 2000] Damianou, N., et al., "Ponder: A Language for Specifying Security and Management Policies for Distributed Systems", Imperial College, UK, Research Report, 2000.
[Damianou, 2001] Damianou, N. et al., "The Ponder Policy Specification Language", Proc. of Workshop on Policies for Distributed Systems and Networks (POLICY2001), Springer-Verlag, LNCS 1995, 2001.
[Garcia, 2005] García, F. J., et al., "Representing Security Policies in Web Information Systems", Proc. of 14th International WWW Conference, ACM Press, 2005.
[Glasgow, 1992] Glasgow, J., et al., "A logic for reasoning about security", ACM Transactions on Computer Systems, Vol. 10, No. 3, 1992.
[Grossi, 2006] Grossi, D., H. Aldewereld, and Frank Dignum. "Ubi lex, ibi poena: Designing norm enforcement in e-institutions." In *Coordination, Organizations, Institutions, and Norms in Multi-Agent Systems II*. LNCS 4386, 107-20. Berlin, Germany: Springer, 2006.
[Guerrero, 2006] Guerrero, A., et al., "Ontology-Based Policy Refinement Using SWRL Rules for Management Information Definitions in OWL", Proc. of 17th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management (DSOM2006), Springer-Verlag, LNCS 4269, 2006.
[Hayton, 1998] Hayton, R. J., et al., "Access Control in an Open Distributed Environment", Proc. of IEEE Symposium on Security and Privacy, Oakland, California, U.S.A, 1998.

[Hoagland, 2000] Hoagland J., "Specifying and Implementing Security Policies Using LaSCO, the Language for Security Constraints on Objects", Ph.D. dissertation, UC Davis, 2000
[Javier, 2006] Javier, R., et al., "A methodological approach toward the refinement problem in policy-based management systems", IEEE Communications Magazine, Vol. 44, No. 10, 2006.
[Jajodia, 1997] Jajodia, S., et al., "A Logical Language for Expressing Authorisations", Proc. of the IEEE Symposium on Security and Privacy, IEEE Computer Society, 1997
[Kagal, 2003] Kagal, L., et al., "A Policy Language for Pervasive Computing Environment", Proc. of IEEE Fourth International Workshop on Policy (Policy 2003), IEEE Computer Society, 2003.
[Keeney, 2003] Keeney, J., et al., "Chisel: A Policy-Driven, Context-Aware, Dynamic Adaptation Framework". Proc. of the 4th IEEE International Workshop on Policies for Distributed Systems and Networks (Policy2003), IEEE Computer Society, 2003.
[Kephart, 2007] Kephart, J.O., et al., "Achieving Self-Management via Utility Functions", IEEE Internet Computing, Vol. 11, No. 1, 2007.
[Ko, 2006] Ko H.J., et al., "A Semantic Context-Aware Access Control in Pervasive Environments", Proc.of the 6th International Conference on Computational Science and Applications **(**ICCSA 2006), Springer-Verlag, LNCS 3981, 2006.
[Loyall, 2011] Joseph P. Loyall, Matthew Gillen, Aaron Paulos, Larry Bunch, Marco Carvalho, James Edmond- son, Douglas C. Schmidt, Andrew Martignoni III, and Asher Sinclair. Dynamic policy-driven quality of service in service-oriented information management systems. *Software: Practice and Experience*, 41(12):1459-1489, 2011.
[Lupu, 1999] Lupu, E. C., et al., "Conflicts in Policy-Based Distributed Systems Management", IEEE Transactions on Software Engineering, Vol. 25, No. 6, 1999.
[Lupu, 2007] Lupu, E., et al., "Autonomous Pervasive Systems and the Policy Challenges of a Small World!", IEEE Computer Society, 2007.
[Lopez deVergara, 2004] López de Vergara, J. E., et al., "Applying the Web Ontology Language to management information definitions", *IEEE Communications Magazine*, Vol. 42, No. 7, 2004.
[Montanari, 2004] Montanari, R., et al., "Policy-based Dynamic Reconfiguration of Mobile Code Applications", IEEE Computer, Vol. 37, No. 7, 2004.
[Moffet, 1993] Moffet, J., et al., "Policy Hierarchies for Distributed Systems Management", IEEE Journal on Selected Areas in Communications, Vol. 11, No. 9, ACM Press, 1993.
[Müller-Scholer, 2004] Müller-Schloer, C. Organic computing " on the feasibility of controlled emergence. In CODES+ISSS '04: *Proceedings of the International Conference on Hardware/Software Codesign and System Synthesis* (Washington, DC, USA, 2004), IEEE Computer Society, pp. 2–5.
[NSA, 2012] National Security Agency (NSA), Enterprise Services Division, Identity and Access Management Branch Digital policy management: A foundation for tomorrow. In RSA Conference. https://ae.rsaconference.com/US12/scheduler/eventcatalog/eventCatalog.do. (accessed November 29, 2012).
[Ostrom, 2008] Ostrom, E. Polycentric systems as one approach for solving collective-action problems. Social Science Research Network, SSRN-id130469, 2008. http://ssrn.com/abstract=1304697 (accessed 18 September 2012).
[OWL, 2004] OWL Web Ontology Language, http://www.w3.org/TR/2004/REC-owl-features-20040210/
[OMG, 2008] The Object Management Group's Semantics of Business Vocabulary and Rules, http://www.omg.org/spec/SBVR/1.0/
[Ortalo, 1998] Ortalo, R., "A Flexible Method for Information System Security Policy Specification", Proc. of the 5th European Symposium on Research in Computer Security (ESORICS 98), Springer-Verlag, LNCS 1485, 1998.

[Phanse, 2006] Phanse, K. S., et al., "Modeling and Evaluation of a Policy Provisioning Architecture for Mobile Ad-Hoc Networks", Journal of Network System Management, Springer-Verlag, Vol. 14, No. 2, 2006.
[Robinson, 1988] Robinson, D.C., et al., "Domains: A new approach to distributed system management", Proc. of the Workshop on the Future Trends of Distributed Computing Systems, IEEE Computer Society, 1988.
[Sen, 2007] Sen, Sandip, and Stéphane Airiau. "Emergence of norms through social learning." Presented at the *International Joint Conference on Artificial Intelligence (IJCAI-07)* 2007, 1507-12.
[Sloman, 1994] Sloman, M. "Policy driven management for distributed systems", Journal of Network System Management, Springer-Verlag, Vol. 2, No. 4, 1994.
[Sloman, 2002] Sloman, M, et al., "Security and Management Policy Specification", IEEE Network, Vol. 16, No. 2, 2002.
[Toninelli, 2007] Toninelli, A., et al., "Proteus: A Semantic Context-Aware Adaptive Policy Model", Proc. of the IEEE Conference on Policy (Policy2007), IEEE Computer Society, 2007.
[Tonti, 2003] Tonti, G., et al., "Semantic Web languages for policy representation and reasoning: A comparison of KAoS, Rei, and Ponder", Proc. of the Second International Semantic Web Conference (ISWC2003), Springer-Verlag , LNCS 2870, 2003.
[Twidle, 2009], Twidle, K., et al., "Ponder2: A Policy System for Autonomous Pervasive Environments", Proc. of the Fifth International Conference on Autonomic and Autonomous Systems, IEEE Computer Society, 2009
[Uszok, 2003] Uszok, A., et al., "KAoS policy and domain services: Toward a description-logic approach to policy representation, deconfliction, and enforcement", Proc. of IEEE Fourth International Workshop on Policy (Policy 2003), IEEE Computer Society, 2003.
[Uszok, 2008] Uszok, A, et al., "New developments in ontology-based policy management: Increasing the practicality and comprehensiveness of KAoS, Proc. of the IEEE Conference on Policy (Policy2008), IEEE Computer Society, 2008.
[Uszok, *et al.*, 2011] Uszok, Andrzej, Jeffrey M. Bradshaw, James Lott, Matthew Johnson, Maggie Breedy, Michael Vignati, Keith Whittaker, Kim Jakubowski, and Jeffrey Bowcock. "Toward a Flexible Ontology-Based Policy Approach for Network Operations Using the KAoS Framework." Presented at the *The 2011 Military Communications Conference (MILCOM 2011) 2011*, 1108-14.
[**van Diggelen, 2009a]** van Diggelen, J., Bradshaw, J. M., Johnson, M., Uszok, A., and Feltovich, P. Implementing collective obligations in human-agent teams using KAoS policies. *Proceedings of Workshop on Coordination, Organization, Institutions and Norms (COIN), IEEE/ACM Conference on Autonomous Agents and Multi-Agent Systems*, Budapest, Hungary, 12 May 2009.
[van Diggelen, 2009b] van Diggelen, J., Johnson, M., Bradshaw, J. M., Neerincx, M., and Grant, T. Policy-based design of human-machine collaboration in manned space missions. *Proceedings of the Third IEEE International Conference on Space Mission Challenges for Information Technology (SMC-IT)*, Pasadena, CA, 19-23 July 2009.
[Verma, 2000] Verma, D. "Policy Based Networking: Architecture and Algorithms", New-Riders, 2000.
[van Sinderen, M., 2006] van Sinderen, M., et al., "Supporting context-aware mobile applications: an infrastructure approach", IEEE Communications Magazine, Vol. 44, No. 9, 2006.
[Woods and Branlat, 2008] Woods, D.D., & Branlat, M. (2008). Basic patterns in how complex systems fail. In E. Hollnagel, Jean Paries, D.D. Woods, & J. Wreathall (Eds.), *Resilience Engineering in Practice* (pp. 127-143). Burlington, VT.: Ashgate Publishing.
[WSPL, 2006] Web Services Policy 1.5 – Primer, http://www.w3.org/TR/2006/WD-ws-policy-primer-20061018

[XACML, 2013] OASIS eXtensible Access Control Markup Language (XACML) https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml
[Xu, 2011] Xu, G., et al., "A Policy Enforcing Mechanism for Trusted Ad Hoc Networks", IEEE Transactions on Dependable and Secure Computing, Vol. 8, No. 3, 2011.