

## Policy Based Management for Internet Communities

Kevin Chekov Feeney, Dave Lewis, Vincent P. Wade

Knowledge and Data Engineering Group

Department of Computer Science

Trinity College Dublin

kefeeney@cs.tcd.ie, Dave.Lewis@cs.tcd.ie, Vincent.Wade@cs.tcd.ie

### Abstract

*Policy Based Management (PBM) is a research topic that has been driven by the tremendous complexity inherent in the administration and management of present-day networking and telecommunications systems and services. The increasingly diverse organisational forms of modern industry represent a significant component of this complexity. Internet communities offer extreme examples of organisational diversity, since they often lack any central authority and many subsections operate with almost complete autonomy. This paper argues that PBM systems offer great potential in this domain due to the complexity of management arrangements. However, since these communities lack any single trusted administrative hierarchy, a centralised solution to policy engineering and management is not possible. This paper proposes an approach to modelling communities for PBM systems. This approach focuses on the concept of communities within a hierarchy of authority as the fundamental unit of organisational analysis. As such, the model reflects the distribution of authority in the real-world community, the resulting policies reflect the community's operational needs and contracts between the various groups and individuals that make up the community. Policy conflicts are used to identify organisational conflicts that must be resolved. In order to illustrate and validate these concepts, the paper presents a conceptual architecture and case study based on the secure management of an open publishing network.*

### 1. Introduction

Policy Based Management (PBM) is currently attracting considerable research focus as an enabling technology for managing large scale, heterogeneous information systems and communications infrastructure [1]. It has the potential to achieve a unified method of managing enterprise information

resources, including such areas as access control, network management (in particular quality of service management) and configuration management. This unified method aims to enable administrators to manage entire networks through the authoring of policy rules (or other elements of a policy language), preferably in a single, simple language. The ultimate aim of a policy-based system is to derive these policies from business goals, so that information systems can respond dynamically to changes in those goals. The motivation being that the resultant management system behaviour is more focused on achieving the business goals of the organisation [2].

PBM rests on the assumption that a common set of rules can be applied to sets of entities in the system, rather than merely to individual entities. Thus, policy languages include constructs to group entities together. This facilitates management by allowing the definition of rules that apply to sets of entities rather than to individuals. Policy rules are divided into authorisation policies for access control management and obligation policies for resource management. PBM systems which include support for both types of policy promise to provide a complete solution to the problem of managing complex electronic networks.

Internet communities often manage large and complex networks of information resources and much of the research into policy specification languages and management architectures is equally applicable in this domain. However, Internet communities have several characteristics which lead to problems in accurately modelling their organisational structures and managing the administration of PBM systems. Most importantly, they tend towards a decentralised, non-hierarchical model of decision making, with resource ownership distributed across the community, rather than being centralised, as is generally the case with hierarchical organisations. Access control models that are commonly associated with PBM systems, such as Role Based Access Control (RBAC) [3], lack this

notion of distributed ownership. They also require the construction of a detailed model of the human organisation. However, the fluidity and heterogeneity of the groupings that make up large Internet communities render centralised approaches to this modelling impossible.

This paper examines the possibilities of applying PBM approaches to the problem of managing the information resources of large voluntary Internet communities. It describes a community policy framework for applying PBM solutions to these communities, a framework which facilitates the distributed, decentralised modelling of the community structure and the building of an organisational structure to reflect the distribution of ownership of the resources managed by the community. It aims to enable the secure management of resources shared between different subsections of the community.

Although this paper particularly focuses on Internet communities, much of the issues addressed are increasingly relevant to traditional hierarchical organisations, where new organisational paradigms, including team-working, virtual organisations and cost-centres share many of these characteristics.

## 2 Internet Communities and Policy

Voluntary Internet communities are a new form of organisation, enabled by mass networked computing. It is difficult to generalise about this type of organisation, due to the diversity of forms that they take. However, we can at least note a few characteristics that they tend to exhibit - especially when contrasted with traditional bureaucratic organisations. They depend upon volunteer labour; their membership and goals can be fluid; they are widely distributed and often all community interaction is electronic; they tend towards flat hierarchies and often have wide membership involvement in decision making; their structures evolve over time; they can be composed of multiple autonomous sub-communities with independent decision making mechanisms and different internal organisations.

There are several well-known examples of such communities such as the open-source software-development community responsible for the Linux operating system. The success of the model has been proved by the widespread adoption of Linux by the IT industry and has prompted a closer examination of this form of working within traditional hierarchical organisations [4].

Although this type of community has flourished in the past decade, there remain several widespread problems to their development. The majority of

projects have remained very small and have had problems in expanding, lacking any means of regulating access to project resources and management responsibilities in a controlled way [5]. They have often remained centred on a single individual maintainer, responsible for all decision making, who becomes a bottleneck to management, thus restricting the expansion of the project. Larger projects, which have succeeded in attracting greater numbers of members, have had problems in integrating the various sub-systems, which work in practice as autonomous projects, require a large amount of manual negotiation and ultimately often depend on a single maintainer to carry out management functions. For these reasons, a PBM approach, with its potential for extensive automation of management tasks, should be particularly attractive to Internet communities.

### 2.1 Case Study: Indymedia Network

This paper will focus on one example of such a community - the Indymedia Network [6], a global open-publishing community - and examine the problems of applying a PBM solution to the management of this community's resources. This community has been selected as its size, complexity and decentralisation provide a good example of the difficulty of applying policy solutions to distributed non-hierarchical organisations. One of the authors has been closely connected to the community for several years and has been involved in developing the Oscailt open publishing system, a software package developed to facilitate management of nodes in the network.

Indymedia is a global open-publishing network that consists of 123 local independent media centres (IMCs) in 52 countries. Each IMC operates as a producer and distributor of locally-produced open-source media content. Although each IMC is autonomously managed, the community is held together by a shared technical infrastructure and a common set of policies, or contracts between each IMC and the rest of the network. The technical infrastructure is developed and managed by a network of working groups and project groups, some of which are sub-communities of a particular IMC and manage local resources, while others - those that are responsible for the development and management of the infrastructure on a regional and global level - draw their membership from across the global network [7].

This complex structure owes little to central planning. It has evolved through a process of specialisation and incorporation, typical of Internet communities [8]. Specialisation takes place as a community divides itself into sub-groupings and delegates specific areas of responsibility to them. For example, IMCs generally start as a single working

group. Over time many have sub-divided into a combination of editorial, video, audio and other groups, each responsible for managing different resources. Incorporation occurs when previously independent groups join the community, bringing resources with them. Thus, many local IMC's were independent projects with a similar aim before being incorporated into the Indymedia community.

The task of co-ordinating management responsibilities and enforcing the various contracts between the working groups requires a significant amount of administrative work and largely depends upon the good faith of the individual administrators of community resources. However, the rapid expansion of the community - from a single IMC with a few dozen members in 1999, to a huge global network involving tens of thousands of people by 2003 and in several thousand working groups - has caused the community to seek automated solutions for enforcing policies. A PBM approach would prove particularly attractive to this community as it offers the promise of automating much of the administration, while also providing an enforcement mechanism for the contracts that bind the groups together.

## 2.2 Policy Engineering for Communities

Modern policy languages and PBM frameworks are technically capable of providing solutions for many of the administrative problems experienced by Internet communities such as Indymedia, as these are largely the same problems encountered when dealing with any large organisation. Working groups could be modelled as roles, domains, groups or some other policy language construct and access control policies could be applied to these, thereby making access control administration significantly more efficient.

Access control models like RBAC, security policy frameworks like Oasis [9], and PBM frameworks like Ponder [10] are based on the assumption that the organisation can be modelled in its entirety, in a centralised requirements engineering phase before deployment. Even when dealing with centralised bureaucratic organisations, this is a difficult task and "the policy development process alone can take months to refine and implement" [11]. However, regardless of the sophistication of the modelling process, there are a number of characteristics of Internet communities which render this entire approach problematic:

- The fluidity of the community's structures mean that by the time the requirements have been captured and modelled as policies, the community structure will inevitably have changed considerably.
- Two work groups with the same overall set of duties and rights with respect to the system may

differ greatly in their internal organisation. Thus, there is no guarantee that useful, reusable grouping abstractions like roles will exist across working groups.

- The internal organisation of the working groups may be private. Although they may participate in a broader community contract and share in the management of community resources, they may not be willing to expose the division of rights and duties within the group
- In hierarchical organisations we can assume that the ultimate authority over the organisation's resources lies at the top of the organisational hierarchy. In Internet communities we can make no such assumption. Many of the shared resources are ultimately owned by autonomous independent groups within the community. Although the administration of these resources may be shared across the community, the groups that own the resources must be able to retain ultimate control over them.

## 3. Community Policy Framework

In this section, we provide the basic outlines of a PBM framework for communities, where the organisational structure of a community is modelled as a set of interrelated communities. Authority over particular resources is distributed throughout the organisation using delegation between communities. This model is intended to reflect the way that these communities work in practice. Rather than creating the structure through a centralised modelling process, the framework enables a policy based approach to be applied to the management of the processes of subdivision, delegation of authority and incorporation through which these communities dynamically define their own structures and operational rules.

Delegation between groups is used to create a hierarchical map of authority over each resource. This authority map also forms the basis for a single integrated model of the community, incorporating users, administrators, managed resources, automatic policy conflict resolution and negotiation, community decision making and controlled management of policy refinement.

For the purpose of this discussion we use the Ponder notation for specifying policy rules. In Ponder authorisation and obligation policies, both positive and negative, are specified as having a subject, action and target which describe who performs the action, what action they carry out and which resource they are acting upon. Obligation policies also include an event which triggers the evaluation of the policy rule, while all policies are also subject to constraints which limit the set of



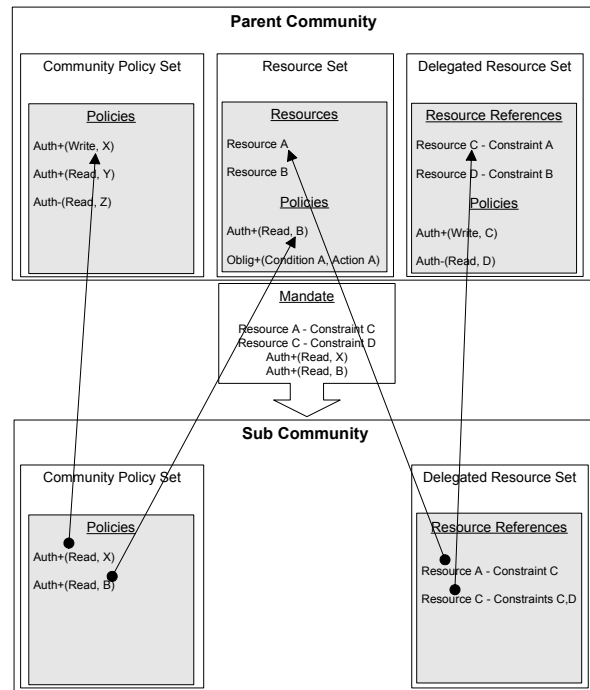
### 3.2 Sub-Communities

At its simplest, an Internet community is modelled as a single undifferentiated top-level community. The modelling of specific groups within the community is achieved through the creation of mandated sub-communities or child-communities, each made up of a subset of the top-level community. Sub-communities can themselves create their own sub-communities. The creating community becomes the parent community of the sub-community. Each sub-community has exactly one parent to ensure a simple hierarchical relationship, or partial order, between communities. Any community can create a sub-community by defining membership rules for it. The sub-community is described as mandated because the parent community may create policy rules whose subject is the sub-community. That is to say that the parent community may define what the sub-community is authorised to do (by defining authorisation policies whose subject is the sub-community) and what it is obliged to do (similarly, by defining obligation policies). The set of policies that have been defined by the parent community and whose subject is the sub-community is the mandate of the sub-community. In certain cases, the parent community will wish to precisely describe the operation of the sub-community, through a detailed and fine-grained definition of its mandate. In other cases, the parent community may specify more general policies in the mandate and allow the sub-community a degree of self-management in terms of refining these policies into more specific policies.

### 3.3 Delegation of authority and rights

Delegation of authority and rights is fundamental to the community policy framework. Any community can delegate authority over any subset of its resources and policies to its sub-communities. If a community has authority over a resource, this means that it can author policies whose target is that resource. When this authority is delegated to a sub-community the sub-community can author policies whose target is that resource. A community can also delegate a subset of its rights to its sub-communities. Delegation of rights means that a community that is the subject of an authorisation policy can pass this authorisation policy on to its sub-communities, effectively making the sub-community the subject of the authorisation policy. The creation of sub-communities and the delegation of authority and rights allows the dynamic creation of an organisational structure for the overall community.

This concept of delegation differs from the concept of delegation in Ponder. Ponder's delegation model allows individuals to nominate others who are trusted to act on their behalf to perform certain tasks, independent of the delegated individual's position in



**Figure 2: Delegation to sub-community**

the organisational structure. We describe this type of delegation as delegation of identity, as the delegated individual acts as if she was the delegating individual, by effectively assuming their identity for the enactment of particular tasks. Delegation of identity is an important feature for any PBM system, as it is a common practice in all organisations. However, it is a different concept to the delegation of authority and rights between communities and the two concepts are entirely compatible within a PBM system.

Practically, delegation of authority amounts to entering a reference to that resource in the set of delegated resources of the sub-community. Similarly a policy that applies to a community can be delegated to its sub-community by inserting the policy rule into the sub-community's set of policies. Negative policies, whether authorisations or obligations, are not explicitly delegated since if a negative policy applies to a community, then it implicitly applies to its sub-communities. Constraints that apply to policies, whether they be meta-policies limiting the form of the policy rule or constraints on the conditions under which a policy is valid, are implicitly propagated from a community to its sub-communities.

### 3.4 Policy Refinement and Resource Semantics

This propagation of authority and rights through the community structure wouldn't be very useful if a community could only delegate precisely those rights that it possesses. However, our model allows for sub-

division of rights in delegation. If we consider a community authorisation policy to have the form  $\text{auth}\{\text{action}, \text{target}, \text{constraints}\}$  (the community is implicitly the subject of the rule), then we can delegate a policy  $\text{auth}\{\text{action}', \text{target}', \text{constraints}'\}$  to a sub-community as long as  $\text{auth}\{\text{action}, \text{target}, \text{constraints}\}$  implies  $\text{auth}\{\text{action}', \text{target}', \text{constraints}'\}$ . For example, if a community has a policy which authorises the writing of files to diskA without any constraints, written as  $\text{Auth}\{\text{write}, \text{diskA}\}$ , then it can delegate the policy  $\text{Auth}\{\text{read}, \text{fileA}\}$  to a sub-community if the semantics of the target resource specify that write authority implies read authority and fileA is a subset of diskA (these semantics are typically dependant on the operating system). Similarly, if the community possesses authority to author policies with targets  $\{A,B,C\}$ , constrained by meta-policies  $\{X,Y,Z\}$ , then the community can delegate authority to its sub-community with meta-policies  $\{X',Y',Z'\}$  as long as the delegated resources are a subset of  $\{A,B,C\}$ . The constraints that apply to policies authored by the sub-community are the union of the existing meta-policies and the new meta-policies applied by its parent, that is  $\{X,Y,Z,X',Y',Z'\}$ . In effect, this propagation of rights and authorities through the community structure means that policies are organically refined as they are distributed through the community hierarchy. As we get further down the community hierarchy, we can expect that policy rules will become more precise and limited in scope as large areas of responsibility are broken down into specific tasks and delegated to various sub-communities.

This process of policy refinement through delegation of rights depends upon the availability of semantic information about the target resource. This semantic information is stored with the resource in the form of a hierarchical tree of actions, relating to the resource. This hierarchical tree provides a partial order on actions, according to the implied relationship, described in [15]. Target resources are also organised in a hierarchical directory structure. This enables the community management system to ensure that all delegations are valid. If a community possesses a right  $\text{Auth}\{\text{ActionA}, \text{TargetA}\}$  then a delegated right  $\text{Auth}\{\text{ActionB}, \text{TargetB}\}$  is validated by ensuring that the right to take ActionA implies the right to take ActionB with respect to TargetA and that TargetA is higher or equal to TargetB in the directory structure.

### 3.5 Decision making and community agents

What do we mean when we say that the community can define policies and that the community is the subject of policy rules? The community is not a single entity and can be made up of many individuals with independent motivations,

who can not be trusted to always carry out the community's will. Thus, we introduce the concepts of community decision-making mechanisms and community agents into our model. Community agents, whether human or automated, carry out actions on behalf of the community. A community

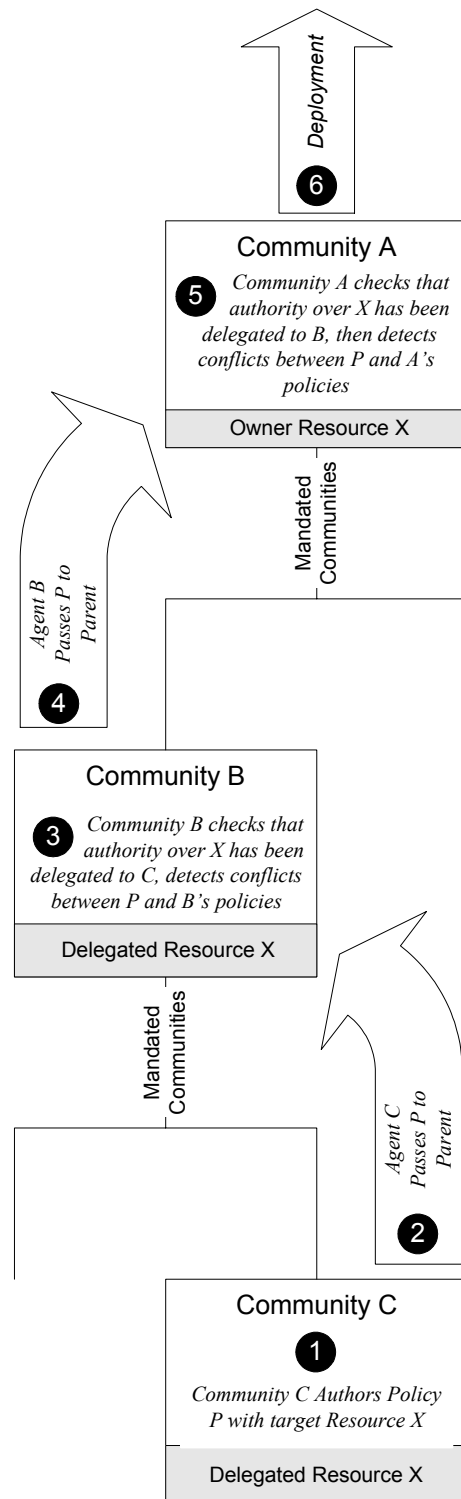


Figure 3: Hierarchical Policy Model

member becomes a community agent when a decision has been made by the community to carry out a specific action. Decision making mechanisms define the process through which a community reaches a decision. This is particularly important when modelling Internet communities as they often operate on a democratic basis. Each community has a default decision making mechanism, but can also attach specific decision making mechanisms to particular policy rules. So, for example, a community may decide that some actions can be taken on behalf of the community by any member of the community, while other actions may require agreement by several members, or even a majority vote, before they can be taken on behalf of the community. Once a decision has been made the community issues a certificate to the community agent which permits the agent to perform the action on behalf of the community.

### 3.6 Internal Privacy and Hierarchies

Authority over resources may be delegated repeatedly from communities to their sub-communities and each may apply policy rules to the resource. However, each community delegates only to its immediate sub-communities and allows them to subdivide the authority between their own sub-communities. Thus communities are only aware of their parent community and their immediate children. This allows communities to keep their internal organisation private from the rest of the broader community. When a community delegates a policy to a sub-community, only an agent of that sub-community can act upon the policy.

Although Internet communities contain many democratic elements, they also contain hierarchies. For example, many users of community provided services may be considered to be part of the community, yet only those who put time into community development may have an input into community decisions. This can be accommodated in our model by specifying a particular sub-community as a control community. A control community acts on behalf of its parent community, that is to say that an agent of the control community is always an agent of its parent community.

### 3.7 Interpreting Policy Hierarchically

Through the process of communities sub-dividing and delegating authority to their sub-communities, a hierarchical tree of communities is built up within the overall community and this amounts to a model of the distribution of authority over each resource managed by the community. This model is used in the delegation process, to ensure that any authority delegated is a subset of the authority possessed. However, it also provides us with a means of enforcing policies in a hierarchical way to automatically resolve policy conflicts.

Even when a parent community delegates authority over a particular resource to a sub-community, the parent may still author policies whose target is the resource. This creates potential policy conflicts on particular resources due to conflicting policies being defined by communities and their sub-communities (which may occur several steps of delegation away). To get around this problem, policy is hierarchically enforced. Thus, policies authored by the community that owns the resource have precedence over policies authored further down the hierarchy. As the hierarchy is based upon authority with respect to the resource, this policy precedence is correct by definition. This is an important feature since it allows us to define community-wide policies at the most general levels, policies that will continue to be enforced irrespective of any policies that are applied in sub-communities. For example, we can impose community-wide security guarantees on all resources in the top-level community while leaving the sub-communities free to define extra policies in their specific areas of responsibility, without the risk that these policies may inadvertently violate the security guarantees.

When a new policy is authored anywhere in the community structure (step 1 in figure 3), it passes up through its parent communities (step 2) until it reaches the owning community of the target resource. At each stage the parent automatically checks to ensure that the new policy rule does not conflict with its existing policies for the target resource (step 3) before issuing a community agent certificate and passing it on to its parent community (step 4). When the rule reaches the owning community, it is deployed to the target resource (step 6). If a policy conflict is detected before deployment, the new policy rule is either rejected or, if possible, automatically rewritten to limit the target domain in order to avoid the conflict. A rejected policy is returned to the authoring community with an indication of the level on which the conflict was detected. The authoring community can then choose to manually rewrite the policy rule to avoid the conflict, or can attempt to negotiate a change in the policy rule at the level where the conflict was detected (which it is by definition a subset of). As policy conflicts can be the result of genuine conflicts between communities, caused by conflicting demands and resource competition, there is no way to avoid all cases of manual re-negotiation of policy rules. In these cases, we need to know at what level of the organisation the conflict must be resolved. The community policy framework automatically detects exactly where the conflict lies within the community structure and the author of the conflicting policy can choose to attempt to negotiate a policy change at the relevant level.

This hierarchical enforcement of policy depends upon a conflict detection mechanism. The discussion of conflict detection is beyond the scope of this paper, for a comparison of various approaches see [16, 17]).

#### 4. Implementation and Case Study

Although it is not directly implementable in any existing PBM system, since they lack community constructs with the required semantics, the community policy framework is not tied to any specific policy language and uses broadly the same policy concepts as other existing PBM systems. Therefore, the approach of mapping the community based model to an existing PBM framework was thought prudent rather than devising yet another Policy language and architecture. The Ponder framework and its suite of tools, along with the source code, are freely available to download on the Internet. Ponder uses a declarative object-oriented language which is specifically designed for simple policy specification - particularly suitable for this domain where policy rules are not authored by experts. Furthermore, Ponder provides a range of grouping constructs and the fact that ponder policies are themselves managed resources allows a great degree of flexibility in designing an administrative model. Therefore, Ponder was used in our experiments as an underlying policy framework.

As a case study and proof of concept, we modelled a simplified version of one branch of the Indymedia Internet community. This community was modelled according to the structure in figure 4. At the highest level, Indymedia is specified as a single community, incorporating all users of the system. This community owns all of the resources that are shared across the community. The global decision making community is the control community of the top-level community, since it makes decisions on behalf of the whole community and has authority over all global resources.

There are other sub-communities of the top-level community, which are delegated authority over subsets of the global resources, such as projects and technical infrastructure. These communities can author policies whose targets are these delegated resources, subject to any constraints that the global decision-making community may impose upon the content of these policies. There is also a European regional community, which is delegated authority from the global community to author policies whose target is the global newswire. For example, they may wish to author a policy forbidding certain types of content from the global newswire. The European community also introduces new resources into the community, namely the European newswire. This is

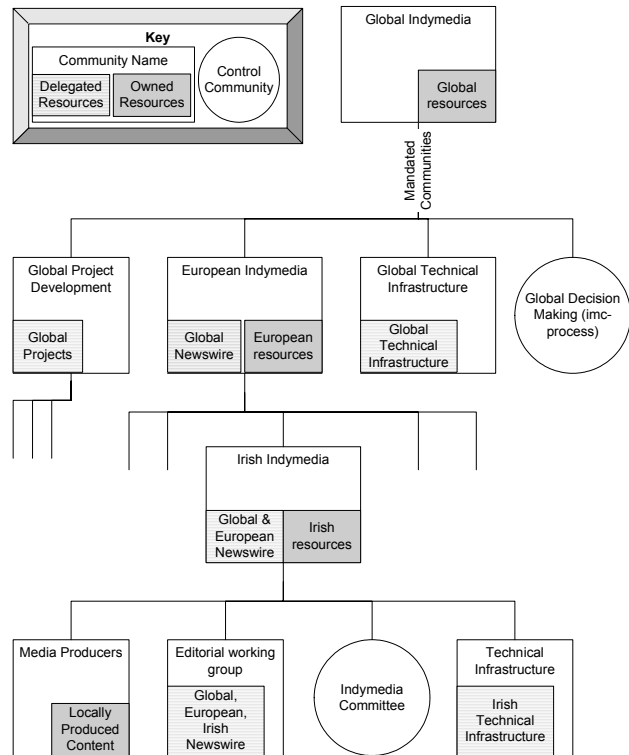


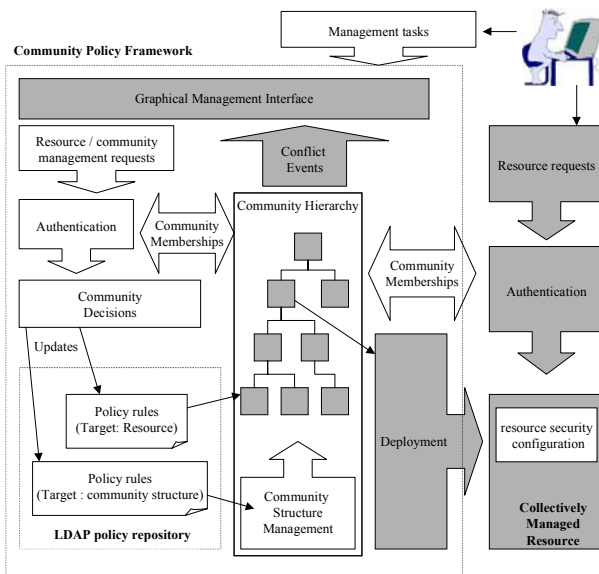
Figure 4: Community Model of Indymedia

autonomously managed from the global community and the global community cannot author policy whose target is this resource.

The Irish Indymedia community is a sub-community of the European Indymedia community. Again it has delegated authority over certain resources that are owned at higher levels and authority over its own resources that it controls autonomously. This authority is further delegated to working groups covering the various areas of the local IMC's operations.

To appreciate the concept of hierarchical enforcement of policy, consider the situation when a member of the Irish editorial working group attempts to access the global newswire. The member must first go through the relevant decision making process before being issued with a community agent certificate to act on behalf of the community. This certificate is then passed to its parent community where it is checked to ensure that access rights for this resource have indeed been delegated to the sub-community. The parent community then issues its community agent certificate and passes it onto its parent where the checks are repeated in turn until the owning community of the resource is reached and access is granted.

When new policies are authored they pass through a similar hierarchical enforcement process. For example if the Irish editorial group wishes to create a



**Figure 5: Community Framework Architecture**

policy banning users from posting images to the European newswire, it would create a policy of the form Auth- $\{Post\text{-image, European Newswire}\}$ . Once this new policy has been ratified by the community's decision making mechanism, it is passed to the Irish Indymedia community where it is checked to ensure that authority over the European Newswire has been delegated to the Editorial Group. Then it is checked to ensure that the policy obeys all the meta-policies that have been imposed by the parent community. Then it is checked to ensure that the policy does not conflict with any of the parent community's policies. If all of these tests are passed, the parent community attaches its agent certificate to the policy and passes it onto its parent where the checks are repeated in turn before deployment.

The community structure is specified in XML, according to the DTD in Appendix 1. The community policy framework allows for each community specification to be stored in different places across the Internet, but for simplicity, the entire community structure was here specified in a single file. This file contains a complete description of the community policy system.

The community structure is interpreted with a custom software application, written in the Java language using the JAXML, JNDI and Ponder libraries. It validates each delegation of authority to ensure that the delegated authority is possessed by the parent community. This is currently achieved by using a simplified hierarchy of actions and target resources that are organised in a hierarchical directory structure. Any constraints placed upon policies by parent communities are also checked.

Currently, the software only detects those conflicts that Lupu describes as modal [17], such as the existence of positive and negative authorisation policies with the same subjects and targets. Once all of these checks are passed, a set of domains, policies and roles to implement the rules of the community structure is created in an LDAP directory, corresponding to Ponder language constructs. The membership rules are evaluated and Ponder User objects are created and assigned domains. Communities are mapped to Ponder domains and a set of policies which dictate the rights and obligations of the members of the domain. The hierarchy of the communities is reflected in the resulting Ponder domain hierarchy. In the current version each member of a community acts as an agent of the community, without decision-making mechanisms, as this concept is not supported by the software framework. Furthermore, the concept of hierarchical enforcement of policy does not fit easily into Ponder's deployment framework, and this has not been implemented in the current version. Nevertheless, the Ponder mapping of the community structure is capable of implementing many of the features of the community model. In particular, the delegation of authority through the community is validated by the software and mapped into a simple set of domains, roles, policies and meta-policies in Ponder.

## 5. Related Work

The Rei policy language [18], which uses RDF-S to specify managed resources, incorporates attribute-based subject credentials from which one could easily build community membership, but it does not support community-based semantics. The language includes detailed delegation semantics, including the ability to restrict the ability for delegates to further delegate authority, which is excluded by our model, where communities have autonomy over delegation of rights and authority to sub-communities. Although the language supports two different types of delegation, while delegation and when delegation, neither of these corresponds closely with the notion of delegation of authority between communities and delegation is seen as an ancillary issue of individual trust rather than as the basis for the organisational model.

Kaos [19], like Rei, incorporates semantic web languages, in this case being entirely specified in DAML. Although it doesn't include any support for community based semantics, the policy framework could be extended by adding the community concept to the ontology. It would be conceivable to implement our community policy framework based on either Kaos or Rei, instead of Ponder as they are all sufficiently expressive and general. However,

none provide native support for community semantics and hierarchical policy deployment and enforcement.

In [20], Wasson and Humphrey describe some of the important issues in implementing policies for Virtual Organisations. However, they assume that policies for shared resources will be specified by centralised administrators and do not consider the possibility that the authoring of policies for resource management could be decentralised and distributed among the bodies who provide the resources.

The Organisation Based Access Control model (Or-BAC) [21] provides means for specifying different security policies for organisations that are structured into several sub-organisations with distinct security policies. However, it does not address the problems of distributed resource ownership or the problem of modelling decentralised communities.

## 6. Conclusion

This paper has presented a community based model for the management of policies in the context of a large Internet community. This model builds upon previous work in the field of Policy Based Management, but simplifies the specification of the organisation by using a single grouping construct, the community, to model the organisation. It introduces a concept of delegation whereby authority and rights are delegated to dynamically build a model of the organisation. Communities retain all of the advantages of specifying policies for roles or domains: policy decisions are made on the basis of the communities that a person belongs to, rather than their identity and policy can be composed through conjunction of the communities that an individual belongs to.

However, communities also extend the standard role-model in several ways. In particular, communities provide a means of incorporating group context and decision making; they integrate administration into the model and allow policy rules to be interpreted with respect to the overall structure of authority in the organisation. They facilitate the self-modelling of groupings within the community, autonomous control of shared resources and allow for privacy of different parts of the structure. A hierarchy of communities provides a means for modelling the organisation in a much more succinct and intuitive way than by using roles with a variety of different relationships between them.

One of the strengths of this framework is that it does not require a detailed process of requirements engineering to create a model of the organisation. We can create the most basic structure and allow the detailed divisions of responsibility and the

organisational groupings to evolve in an organic manner. Thus, for example, we can introduce a PBM system by merely modelling the entire organisation as a single community which is responsible for the full set of resources to be managed by the system. As the needs arise, we can create sub-communities and delegate to them responsibility for specific resources. The structure of the organisation can remain in constant flux, as we strive to more fully understand the key rules of its operation. The analysis of policy conflicts can be used to signal structural problems in our model and in the underlying real-world community, thus giving us constant feedback and impetus to refine our model.

By implementing a community policy framework, based on the Ponder framework, we have demonstrated the feasibility of implementing a community structure in a general policy language. However, there are particular features of the community model, notably the hierarchical enforcement of policies, that are not supported by available PBM systems. Therefore, these concepts have yet to be tested in practice.

Although this paper has specifically looked at the problem of managing policies in large Internet communities, the model is not restricted to this domain. The Internet paradigm, of autonomous groups co-operating as part of larger organisations, is increasingly being employed by traditional organisations. SME value chains, collaborative projects between corporations, virtual organisations and the practice of corporate divisions being run as independent businesses all conform to this paradigm to a certain extent. All of these types of organisation encounter the same problems as an Internet community in relation to the management of shared resources. They need a way to distribute authority without sacrificing ultimate control over their resources. The community based model could prove

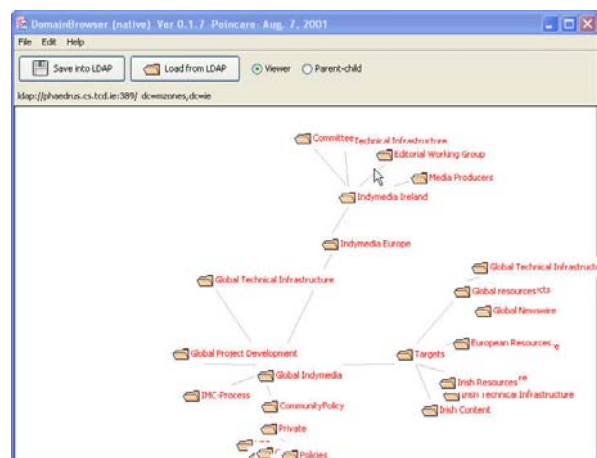


Figure 6: Ponder Domain Structure

to be a promising model for introducing a PBM approach in these forms of organisation.

## 7. Further work

We are currently introducing the community policy framework to manage certain areas of the Indymedia Internet community mentioned in the paper. The system will be introduced to first mirror and then directly manage a particular resource currently managed by a set of working groups. The resource is a CVS repository which contains the development code for one of the community's content management systems. The community will create sub-communities and delegate various authorities to them. For example, there will most likely be a 'developer' working group which will have the authority to add files to the repository, while the 'tester' workgroup will only be permitted to add files to certain branches. The experience of using this PBM approach in a real-world community will provide us with further proof of the concept as well as empirical evidence relating to the usefulness of this management solution. Metrics such as administrative work hours, volume of administrative communication and administrator response times will be measured through user-surveys and monitoring and analysis of communication channels such as mailing lists, to evaluate the usefulness of the approach. Simultaneously, the system will be expanded gradually to manage a larger number of the community's resources, with the aim of eventually providing a universal solution to the community's management needs.

Though the approach presented provides an approach to resolving application specific policy conflict by alignment with the natural decision making processes of an organisation, it does not address the more general problem of mapping high-level user objective to system level objectives. Policy languages in general fail to address this problem because of the shallowness of the semantic model of the managed resource that is incorporated into these frameworks.

Our approach also introduces a variation of this problem, when we wish to get automated assistance in ensuring the community-generated policies are correctly mapped from the community mandate within the constraints of the relevant resource semantics. A likely solution to this problem is to employ ontologies as explicit mechanisms for describing resource semantics. The growing popularity of this approach in the Semantic web may ensure that more and more managed resource will be available with accompanying semantic mark-up. The Rei and Kaos projects indicate moves towards this approach by using semantic web languages - RDF

and DAML respectively, to express policies, though broader integration with existing ontologies is yet to be demonstrated.

We will examine the use of ontology languages for defining our communities, resources and community policies. We aim to benefit not only from better policy mapping due to more accurate resource models, but also from the availability of a wide range of ontology-compatible inference and rule engines to provide policy enforcement, mapping and conflict detection support. We will also investigate the merits of constructing policy rules from a specialisation of the RuleML semantic rule language being proposed to the W3C.

## References

- [1] Wright, S., Lapiotis, P., Chadha, R. "Policy-Based Networking", IEEE Network, Vol. 16, no. 2, 2002, p 8-9.
- [2] Sloman, M., Lupu, E. "Security and Management Policy Specification", IEEE Network, Vol. 16, no. 2, 2002, p 10-19.
- [3] Sandhu, R.S. et al., "Role Based Access Control Models", IEEE Computer, vol. 29, no.2, 1996, p 38-47
- [4] Thompson P. and McHugh, D., Work Organisations, Palgrave, New York, 2002.
- [5] Krishnamurthy, S. "Cave or Community?: An Empirical Examination of 100 Mature Open Source Projects" First Monday, vol. 7, no. 6, June 2002, Available from: [http://firstmonday.org/issues/issue7\\_6/krishnamurthy/](http://firstmonday.org/issues/issue7_6/krishnamurthy/)
- [6] Hyde, G., "Independent Media Centers: Cyber-Subversion and the Alternative Press", First Monday, vol. 7, no. 4, April 2002, Available from: [http://www.firstmonday.dk/issues/issue7\\_4/hyde/](http://www.firstmonday.dk/issues/issue7_4/hyde/)
- [7] See Indymedia Documentation Project, Global Section. <http://docs.indymedia.org/view/Global/ImcProcess>. Accessed November 2003.
- [8] Iannacci, F. "The Linux managing model", First Monday, vol. 8, no. 12, December 2003, Available from: [http://www.firstmonday.org/issues/issue8\\_12/iannacci/](http://www.firstmonday.org/issues/issue8_12/iannacci/)
- [9] Bacon J., Moody K. and Yao W. "A model of OASIS role-based access control and its support for active security", ACM Transactions on Information and System Security (TISSEC), Vol.5, No. 4 (November), pp 492-540, ACM Press, New York, NY, 2002.

- [10] Damianou, N., "A Policy Framework for Management of Distributed Systems", PhD Thesis, Imperial College, University of London, 2001.
- [11] Jude, M., "Policy-based Management: Beyond the Hype" Business Communications Review, March 2001, p 52-56.
- [12] ISO/IEC 15414:2002 JTC1/SC7 Information technology - Open Distributed Processing - Reference Model - Enterprise Viewpoint, ISO/IEC 2002
- [13] Sandhu, R. et al. "The ARBAC97 model for role-based administration of roles: Preliminary description and outline", Proceedings of 2nd ACM Workshop on Role-Based Access Control, Fairfax, VA, November 6-7 1997.
- [14] Hine, J., Yao, W., Bacon, J. and Moody, K.. "An Architecture for Distributed OASIS Services", Proceedings of Middleware 2000, New York, USA, Lecture Notes in Computer Science, Springer-Verlag, 4-8 April 2000. pp 107-123.
- [15] Shen, H., Dewan, P. "Access Control for Collaborative Environments", Proceedings of the ACM Computer-Supported Cooperative Work Conference, Toronto, Canada 1992, pp 51-58
- [16] Tonti, G., Bradshaw, J.M., Jeffers, R., Montanari, R., Suri, N., Uszok, A., "Semantic Web Languages for Policy Representation and Reasoning: A Comparison of KAoS, Rei, and Ponder" Proceedings of 2nd International Semantic Web Conference (ISWC2003), October 20-23, 2003, Sanibel Island, Florida, USA
- [17] Lupu, E.C, Sloman, M. "Conflicts in Policy-Based Distributed Systems Management", IEEE Transactions on software engineering, vol. 25, no. 6, November 1999. pp 852-69.
- [18] Kagal, L., Finin, T., Joshi, A., "A Policy Language for a Pervasive Computing Environment" Proceedings of IEEE 4th International Workshop on Policies for distributed Systems and Networks, June 2003 Lake Como, Italy p 63-77
- [19] Uszok, A., Bradshaw, J., Jeffers, R., Suri, N., Hayes, P., Breedy, M., Bunch, L., Johnson, M., Kulkarni, S., Lott, J., "KAoS Policy and Domain Services: Toward a Description-Logic Approach to Policy Representation, Deconfliction, and Enforcement" Proceedings of IEEE 4th International Workshop on Policies for distributed Systems and Networks, June 2003 Lake Como, Italy p 93-99
- [20] Wasson, G., Humphrey, G., "Towards Explicit Policy Management in Virtual Organizations" Proceedings of IEEE 4th International Workshop on Policies for distributed Systems and Networks, June 2003 Lake Como, Italy pp 173-182.
- [21] El Kalam, A.A, Benferhat, S., Miège, A., El Baida, R., Cuppens, F., Saurel, C., Balbiani, P., Deswarte, Y., Trouessin, Y., "Organization based access control" Proceedings of IEEE 4th International Workshop on Policies for distributed Systems and Networks, June 2003 Lake Como, Italy p 120-135

## Appendix 1: XML DTD & Code Snippet

```

<?xml version="1.0" encoding="ISO-8859-1" ?>
<!ELEMENT communityStructure (community)>
  <!ATTLIST communityStructure location CDATA #REQUIRED>
  <!ATTLIST communityStructure storageType (LDAP|RDB) "LDAP">
  <!ATTLIST communityStructure storageBase CDATA #REQUIRED>
<!ELEMENT community (membershipRules, resourceSet?, delegatedResourceSet?, controlCommunity?,
community*, policyRules?)>
  <!ATTLIST community name CDATA #REQUIRED>
<!ELEMENT membershipRules (policyRules?)>
<!ELEMENT policyRules (rule+)>
<!ELEMENT resourceSet (resource+)>
<!ELEMENT resource (policyRules)>
  <!ATTLIST resource name CDATA #REQUIRED>
  <!ATTLIST resource resID ID #REQUIRED>
<!ELEMENT delegatedResourceSet (resourceReference*)>
<!ELEMENT resourceReference (policyRules)>
  <!ATTLIST resourceReference resIDref IDREFS #REQUIRED>
  <!ATTLIST resourceReference targetSubsetRef CDATA #REQUIRED>
<!ELEMENT controlCommunity (resourceSet?, community*, membershipRules,
controlCommunity?,delegatedResourceSet?)>
  <!ATTLIST controlCommunity name CDATA #REQUIRED>
<!ELEMENT rule (#PCDATA) >
  <!ATTLIST rule language (PONDER|OTHER) "PONDER">

<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE communityStructure SYSTEM "policyCommunity.dtd">
<communityStructure location="phaedrus.cs.tcd.ie:389"
storageType="LDAP" storageBase="dc=mzones,dc=ie">
<community name="Global Indymedia">
  <membershipRules>
    <!-- Membership rules here -->
  </membershipRules>
  <resourceSet>
    <!-- Note: the community management software is implicitly included in the resource list
-->
    <resource name="This" resID="This">
      <policyRules> <!-- policy rules with the community structure as target -->
        <rule><!-- Ponder Rule Goes Here -->
        </rule>
      </policyRules>
    </resource>
    <resource name="Global Resources" resID="GLOB"> <!-- resources refer to LDAP directory -
->
      <policyRules> <!-- policy rules with the particular managed resource as target -->
        <rule>
        </rule>
      </policyRules>
    </resource>
  </resourceSet>
  <controlCommunity name="IMC-Process">
    <membershipRules>
    </membershipRules>
  </controlCommunity>
</community name="Global Project Development">
  <membershipRules>
  </membershipRules>
  <delegatedResourceSet>
    <resourceReference targetSubsetRef="Global Projects" resIDref="GLOB">
  <policyRules>
    <rule>
    </rule>
  </policyRules>
  </resourceReference>
</delegatedResourceSet>
</community>
</community name="Global Technical Infrastructure">

```