Autogeneous Authorization Framework for Open Access Information Management with Topic Maps

Robert Barta[1]; Markus W. Schranz[2]

[1] Austrian Research Centers Seibersdorf
Seibersdorf, Austria
e-mail: robert.barta@arcs.ac.at
[2]Department of Distributed Systems, Institute for Information Systems, Vienna University of Technology
Argentinierstr. 8/184-1, Vienna, Austria
e-mail: schranz@infosys.tuwien.ac.at

Abstract

Conventional content management systems (CMSes) consider user management, specifically authorization to modify content objects to be orthogonal to any evolution of content within the system. This puts the burden on a system administrator or his delegates to organize an authorization scheme appropriate for the community the CMS is serving. Arguably, high quality content - especially in open access publications with little or no a priori content classification – can only be guaranteed and later sustained, if the fields of competence of authors and editors parallel the thematic aspect of the content. In this work we propose to abandon the above-mentioned line of demarcation between object authorization and object theming, and describe a framework which allows to evolve content and its ontological aspect in lockstep with content ownership.

Keywords: Ontology; Semantic Technologies; Authorization Framework

1. Introduction

Content ownership, joint or individual, is the main driving factor in an information society. Currently systems tend to be built with strong gravitational forces to attract content creation, so that the harvested information can be sold back into society. In the long run such business models can lead to monopolies and to highly uneven content distribution.

Traditionally, user management has been regarded orthogonal to the life cycle of a document object within a content management system (CMS). That has allowed implementations to delegate not only authentication but also authorization to a middleware layer. Many modern platforms (such as .NET or J2EE) allow to deploy a wide variety of authorization technologies.

Most CMSes provide an identity-based authorization scheme to control access to the information nodes within the CMS. Individual users either get assigned particular privileges relative to these nodes, or particular privileges are usually clustered into ‘roles’, mainly to reduce the management effort. When particular users are associated with a particular role, they inherit all the role’s privileges. Role management is then usually handled by an administrator. Such an individual person can easily present a bottleneck (and a security risk), so role assignment is often delegated, or even further subdelegated.

Current authorization schemes are quite flexible and they can cover systems in the Wiki class (rather flat user base, hardly any workflow) up to corporate systems with a rather deep organizationally imposed group hierarchy and considerable workflow capabilities. Despite the delegation features in many practical deployments user management is still funneled through very few administrators. And in many real world deployments these actually do not take part in the collective authoring effort.

On a different front, a more recent trend in CMSes is the addition of semantic information (e.g. [1,2]). In the simplest case this is achieved by providing a background taxonomy against which the information nodes are organized. More sophisticated CMSes allow users to attach not only meta-information along well-defined attributes but also to use one of the semantic web technologies (RDF[4] and Topic Maps[5]). These enable to freely relate information items within the CMS against each other or with concepts and instance data defined elsewhere. Such outside information can be either referenced or integrated via virtualization[6]. Also here systems differ considerably in the degree how individual users can extend the existing ontology or the types of relationships.

In our approach we propose to coalesce the authorization mechanism with the ontological information, thus offering an Autogeneous Authorization Framework (AAF) for (open access) information management based on Topic Maps. The paper is structured as follows: in section 2 we describe the challenges for integrating content management and authorization, introduce our proposed methodology and refer to related work and necessary notation formats. In the following sections we formalize our proposed machinery for the AAF and cover implementation aspects such as visibility rules for nodes and the necessary ontological commitments. Finally we summarize our current experiences and outline the work to invest for a scaleable implementation.

2. Challenges in Combining Content Management and User Authorization

As target audience for the integration of content management and authorization we have in mind loose federations of organizations which want to cooperate in certain areas on a number of topics. Each of the involved organizations may have their experts in certain areas but each will seek expert knowledge from their partners.

2.1 Proposed Method and Assumptions

Realistically such federated projects will not have stable ontologies from the very beginning, much in contrast to a priori created ones (e.g. [3]). These will have to evolve over time and each snapshot implicitly indicates deficits and hence the need on which topics the project will have to focus next. This will prompt experts to adopt certain topics and detail them to the extent necessary or feasible.

From past experiences it can also be expected that further field experts will be solicited, adhoc or via affiliation, especially in the area of open access research publications, where major focus lies on content quality and reliability and trust in topic experts. Ideally these invitations for co-authorship will not affect the whole content body but only that fraction for which a new expert is authoritative.

Under the regime of 'taxonomy-based authorization' a particular user does not derive his privileges relative to a given node from the settings of a central administrator or a membership in a group, but from the commitment that the user is an expert in the field the node belongs to (the 'theme'). Accordingly, the system tracks for each node how it is classified to a theme in the current taxonomy and it also keeps book which individuals are authoritative in certain themes.

From there we generalize the regime in several directions:

- First we abstract away from the particular privileges an individual may have regarding a particular topic. In the simplest case this may be a read or write (edit) access to a node. In more complex cases privileges may include the start, promotion or finalization of workflows steps, or different level of read access to only certain fractions or aspects of the topic.
  The only thing we are currently assuming is that all different privileges are totally ordered,

so that (a) it is always unambiguously determinable which privilege is higher than the other, and (b) there is always a highest privilege.

- When topics are generated, they will be classified into a theme. From there topics themselves may or may not follow a workflow. As usual with workflows, the progress of a workflowing node depends on certain privileges, be it for promoting the topic into a final stage or be it for sending it back to an editing phase. Our framework does not prescribe any particular workflow states or any intrinsic privileges to move a topic to another state. The only assumption here is that any workflow privileges are still tied to themes and that topics can be adopted by anyone who has the sufficient privileges for that theme. Topic adoption can happen either actively by a user (pull), or passively (push) by forcing adoption upon the user by another one with higher privileges.
- Also privileges themselves will follow a life cycle, in that the initial privileges of users are extended (monotonically increasing over time). Also here we allow a push-pull setup: either a user, unsolicited, gets privileges via other users to certain themes, or a user requests higher privileges, which he later is possibly granted.

Hereby we allow two subschemes:

- In the 'delegation scheme' privileges can only be granted from someone with higher privileges on that theme.
- In the 'peer scheme' the granted privilege can be at the same level as that of the granting user.

The ontology-based authorization described here can be bootstrapped from any existing taxonomy, even a pathological one with a single theme (the 'thing'). Authors in the system can create taxonomy nodes along with information nodes and establish the highest available privilege on that taxonomy node for them. Safeguards are in place to avoid that users can reassign nodes in the taxonomy to subvert the authorization system.

2.2 Basic Requirements and Related Standards

Since basic elements in our AAF are denominated above as *nodes* in graphs that can be accessed within certain actions, we propose a standardized notation format to represent the resulting semantic network. In literature such graph structures have been implemented in various forms under different names including *associative nets, semantic nets*, *partitioned nets*, or *knowledge maps* in many AI systems. One of the most completely worked out notations are the conceptual graphs developed by Sowa et.al.[8].

Semantic networks rely on a basic model that is similar to that of the topics and associations found in indexes. Thus the two approaches promise great benefits in both information management and knowledge management. Exactly these benefits are targeted at by the topic map standard. By introducing relations between topics and occurences additionally to the topic-association model, topic maps provide a means bridge the gap, as it were, between knowledge representation and information management. In this paper we want to extend this basic intension of topic maps to include user authorization based on thematic topics within the contents.

2.2.1 Topic Maps and the Topic Map Standard

Topic maps are an ISO standard[9] for describing knowledge structures and associating them with information resources. Since topic maps are often synonymed as the GPS of the information universe,

they provide powerful new ways of navigating large and interconnected information sets. According to the basic elements of informations structuring and accessing embodied in indexes, the topic map standard is based on Topics, Associations, and Occurrences. The following section outlines the TAO of topic maps, as it is explained in detail by S. Pepper in [10]. We will use topic maps as notation standard for our AAF as described in section 4.2.

Topics

A *topic*, in its most generic sense, can be any *thing* whatsoever — a concept, an article, person, etc.. The term *topic* refers to the object in the topic map that represents the subject being referred to. Typically, there is a one-to-one relationship between subjects and subjects, with every topic representing a single subject and every subject being represented by just one topic.

In a topic map, any given topic is an instance of zero or more *topic types,* thus categorizing specific topics according to their kind. Similar to the usage of multiple indizes in a book (index of abbreviations, names, or illustrations) topic types semantically describe the nature of topics. What one chooses to regard as topics in any particular application may vary according to the needs of the application, the nature of the information, and the uses to which the topic map will be put: e.g. in software documentation they might be variables, functions, and objects.

In order to identify objects symbolically, topics may have explicit names. Since names exist in various shapes, such as formal names, symbolic names, nicknames, etc. the topic map standard provides the facility to assign multiple base names to a single topic, and to provide variants of each base name for use in specific contexts.

Occurrences

A topic may be linked to one or more information resources that are somehow relevant to the topic. Such resources are named occurrences of the topic and are generally external to the topic map document itself, and they are referenced using various mechanisms the system supports, e.g. URIs in XTM[7]. A significant advantagesto using topic maps is that the real world documents (occurrences) themselves do not have to be touched and thus topic maps support a clear separation of the network into two layers of the topics and their occurrences.

Following the concepts in the topic map standard, also occurrences may be of any number of different types (e.g. *articles*, *monograph*, *commentary*) . Such distinctions are supported in the standard by the concepts of occurrence role and occurrence role type. These basic constructs refer to the basic organizing principle for information. The concepts of *topic*, *topic type*, *name*, *occurrence,* and *occurrence role* provide means to organize information resources according to topics/subjects, and to create simple indexes.

Associations

What we really need in addition to basic indizes for constructing semantic networks is to be able to describe relationships between topics. To achive this, the topic map standard provides a construct called the topic association, which asserts a relationship between two or more topics.

Similar to the grouping of topics and occurrences according to their specific type – such as author/ research area and article/commentary/monograph - also associations between topics can be categorized according to their type. And following the notation concepts of the standard, association types are themselves defined in terms of topics. The ability to apply typing to topic associations significantly increases the expressive power of the topic map, making it possible to group together the set of topics that have the same relationship to any given topic. This is necessary to provide intuitive and user-friendly interfaces for navigating large information networks.

Each topic that participates in an association plays a role in that association. Consequently, association roles can be typed and the type of an association role is also represented as a topic. Due to the fact, that associations in topic maps are multidirectional, the clear distinction between specific association roles is very important (e.g. CM_user *has_edit_right_on* article).

The clear separation of (real world) information resources and the topic map itself, the same topic map can be overlaid on different information sets. Similarly, different topic maps can be overlaid on the same pool of information to provide different *semantic views* to different users. Furthermore, this separation provides the possiblity to interchange topic maps among publishers or to merge several topic maps, thus handling semantic networks.

Omitting other details of the topic map standard since out of scope of this paper we progress to introduce a notation scheme for the AAF, followed by a model on how to represent the nodes in Topic Maps.

3. Proposed Concepts for an Autogeneous Authorization Framework

To describe and analyze the dynamics of an AAF driven system, we need to abstract away (a) from any specifics of the underlying CMS and (b) from any representation technique used to manifest AAF-related ontological and operation information. For this purpose we introduce a simple adhoc formalism to describe static and dynamic integrity constraints.

As minimal ontological commitment we choose to have *nodes* as the unit to carry content. From the AAF's point of view such a node itself is atomic: It can carry any content, be it text, structured or unstructured. The node may have attachments, or it may have meta information attached to it; in any case this is outside the scope of AAF.

3.1 Static Model

3.1.1 Themes

One special node kind are /themes/. Intuitively they represent topics such as finance or, say, UMTS. Themes can be organized into subclass relationships, usually referred to as taxonomy. We write

$$t' < t$$

if the theme t' is a specialization, direct or transitively, of theme t.

Of course, themes can be related to each other in more specific ways, but this is outside the AAF realm. The only exception are non-theme nodes which are affiliated with a theme t, something we denote as

$$n \rightarrow t$$

Any number of such affiliations may exist at any time.

How such affiliation is modeled in the background ontology is deployment and implementation dependent. Here the notation should simply transport that the node *is somehow related* to a certain theme.

One constraint imposed is that such affiliation inherits downwards the subclass hierarchy, so

$$n \rightarrow t \Rightarrow n \rightarrow t', \quad \forall\, t' < t$$

3.1.2 Users

Another special node type are `users`. Nodes of this type are supposed to carry content about a certain user and we implicitly identify a user with its node (which is somewhat sloppy from an ontological view point).

Users are the only active component, so it is them who perform actions.

3.1.3 Actions

AAF also assumes that there is a finite set of actions on nodes. Built-in actions are *`read`* and *`edit`*. *`read`* will always keep the content of the node, *`edit`* will always modify the node, but both will maintain the identity of any node.

Additionally applications and deployments are free to add workflow actions, i.e. actions which move nodes through a series of workflows. Formally, *read* and *edit* are embedded in this scheme as they are the only actions in their respective workflows.

As common in workflow applications every workflow step will move the document into a new state, such as "edited" after an *`edit`* action. States are regarded here as derivatory concepts; still we conveniently

use the notation

n @ S

when a node n is in state S.

On any set of action we also impose an order, so that between two actions there can be a comparison, which of them is *stronger*. This is to model that usually editing implies also reading, or that moving a document in a workflow also implies editing it.

As this comparison may only exist on some pairs of actions, we only need a half-order. In any case we require that there is a strongest action. We refer to it as *`top`*. The *`bottom`* action also exists in every system and it represents the empty action. All actions are bigger that bottom.

3.1.4 Privileges

When users have privileges then these are characterized by the maximal action that user can exert on a node affiliated to a certain theme. We denote such a privilege of user u relative to a theme t for action a as

p ~a~> t

An example would be that Bill has editing rights for *`finance`*:

bill ~edit~> finance

Also here we expect the privilege to *inherit* downwards the theme taxonomy:

u ~a~> t ⇒ u ~a~> t' with t' < t

This makes sense as if Bill has editing privileges for finance he implicitly should also have one for accounting. But since actions are also under a half-order more privileges can be inferred as well:

u ~a~> t ⇒ u ~a'~> t with a' < a

In the example we derive from Bill's authorization to edit finance topics also his authorization to read them. Additionally we define that arbitrary nodes n affiliated to a theme t can be actioned too:

$$\mathbf{u \sim a \sim> n \quad iff \ \exists\ t : n \rightarrow t \ and \ u \sim a \sim> t}$$

If a node is not (yet) affiliated with a theme, then there is no access information for it to be inferred.

In the same way as document states can be derived from the actions, user roles can be defined on the basis of their privileges. Since all privileges, though, are always relative to a theme, simply to define that someone is an *editor* is correct, but ultimately looses essential information:

$$\mathbf{Editor\ (u) \Leftarrow \exists\ t:\ u \sim edit \sim> t}$$

Still we keep this as notational convenience.

3.2 System Dynamics

According to the set of privileges at a given time, nodes can evolve and move through their respective workflows. Privileges can also change throughout the lifetime of an AAF governed system. This is either achieved by extending someone's privileges directly, or indirectly in that nodes are affiliated with themes someone has a privilege on.

The integrity constraints for this evolution we model with the help of pre- and post-conditions on node transitions. The preconditions guard certain actions and the post-conditions characterize the state in terms of AAF after a node has be acted on. Each of these transactions are atomic.

To denote, for example, that every user is entitled to edit his node u into a new version u' we write

$$\mathbf{< User\ (u),\ u \sim edit \sim> u \ ||\ User\ (u')\ >}$$

While the node u undergoes a change, it will maintain its identity.

3.2.1 Bootstrapping

To put an AAF system into an initial state, it has to be bootstrapped into some configuration. The most minimal state is characterized by

$$\mathbf{superuser \sim top \sim> thing}$$

Superuser is one particular user. What makes him special is that he has the highest privilege (`top`) on the most abstract *thing*.

3.2.2 Document Life Cycle

When a new document node is created it will not have any affiliation. Such an *outlawed* node can only be brought into the realm of a theme t if a user u has top privileges on t:

$$\mathbf{< Node\ (n),\ u \sim top \sim> t \ \|\ n \rightarrow t' > \qquad with\ t' \leq t}$$

In this initiation phase the user has significant responsibility to choose the smallest reasonable subtheme `t'` of `t`. Further changes of affiliations can happen later as well, but only in an accumulative manner, so that no existing privileges are hampered.

If a node is moved along a workflow axis it always maintains its identity, even when it is modified. A user

u can exert action a on a document node n in workflow state S when a theme t exists such that:

$$\mathbf{< n \rightarrow\ t, n\ @\ S\ , u\ \sim a\sim>\ t\ \ ||\ \ n' >}$$

For the special, predefined actions *read* and *edit* we define

$$\mathbf{< n \rightarrow t, u\ \sim read\sim>\ t\ \ ||\ \ n >}$$

$$\mathbf{< n \rightarrow t, u\ \sim edit\sim>\ \ t\ \ ||\ \ n' >}$$

3.2.3 User Life Cyle

When a user node is created obviously no explicit privileges are defined for that user. Implicitly we allow users to modify their own nodes, mainly to introduce themselves to the swarm of other users. This can be achieved on the policy level with priming every user node `u` with

$$\mathbf{u\ \sim edit\sim>\ u}$$

During the course of the life time a user can acquire new privileges whereby we distinguish two scenarios:

- In the unsolicited scenario a user will be promoted by another user without prior request:

  < User (u), User (v), v ~a~> t || u ~a'~> t' >

  In any case `t'` will be equal or smaller than `t` if the user `v` determines that `u` only needs privileges for a more special theme. "The granting v may also reduce the action level itself so that a' ≤ a .
  If `a` = `a'` we call this process peer invitation, otherwise delegation.

  Following our running example, the editor of the finance sector may hand down reviewing rights to another person:

  < User (bill), User (fred), fred ~edit~> finance || bill ~edit~> accounting >

- Solicited privilege escalation is not likely to be agile. The process requires that users constantly monitor for the needs of other users. This is not something humans are well equipped to do.

  In the solicited scenario a user first requests certain privileges. These will be responded to at a later point by other users, so that requests are then resolved.

  To better moderate the process of solicited privilege escalation we force users either to escalate along the theme taxonomy or alternatively along the action half-order. For the latter case we characterize the creation of a privilege request via

  < User (u), u ~a~> t || u ~b?~> t >

  With `u ~b?~> t` we symbolize the fact that u wants privilege to do `b` on `t`. Note that `u` needs to have at least privilege `a` to launch such a request.

  To escalate along the taxonomy also the user needs minimal entry rights:

< User (u), u ~a~> t || u~a?~ t' >

with `t < t'`.

Every request can be responded to. In the case a request is granted, another user with sufficient privileges will have to come into play:

. **< u ~b?~> t', v ~c~> t" || u ~a~> t >**

Hereby v can choose to reduce a to b, so that $a < b \leq c$. The user v can also choose to reduce the scope of the privilege, so that $t < t' \leq t''$.

If a is chosen to be *bottom*, i.e. the smallest action, then effectively the request is rejected.

4. Implementation

4.1 Visibilities of Nodes Aspect

Once an AAF system is implemented, the user interface has to control which aspects of a node are visible to whom. Many of these visibilities will be policy controlled, so the following may be vary between deployments.

Regardless of the node type we distinguish between the content itself, the ontological embedding of the node and the defined (or derived) privileges on it. For reasons of reproducibility and auditing not only the current information is displayed but also the past history of changes, so that it is imminent who got which privilege at which time.

While general document nodes follow the generic rules of section 3, user and theme nodes have to be treated differently.

4.1.1 User Nodes

For user nodes the content visibility follows the generic rules above. Ontology related information cannot be changed after a user node has been created, at least as far as AAF is concerned. Any ontological content is normally visible to everyone else.

Typically all user s also see all existing privileges of another particular user. *Outgoing* privilege requests, so those which are pending, will be listed only for that particular user and for those users who have a stake in the themes involved and where their privilege level is at least on par with the one requested. Otherwise the request will normally not be shown.

Incoming privilege requests, i.e. those where a particular user has the privilege level to grant or reject the request are listed for that very user. As a convenience this list will include past grants and rejections.

4.1.2 Theme Nodes

Again for the content itself the generic AAF rules apply. As themes are meant to be abstract, only their relationship with other themes can be modified. Every theme node will list the privileges on them, direct or derived.

4.2 Model Representation with Topic Maps

All AAF-related information can be mapped into a topic map, although auditing information recording events is less suitable to be brought into this representation. That will be kept normally separate.

The initial topic map will have to contain the ontological commitments, namely that every node is either a general document node, a user node or a theme node:

user subclasses **node**

theme subclasses **node**

The predicate `User(u)` is then true when u is a user node in the map. Similarly this holds for themes.

Another commitment is the list of actions involved. The predefined ones will appear in any case:

read isa **action**

edit isa **action**

But more can and should be readily added. Any ordering between actions is represented by an association of type *comparison*

comparison (stronger: **edit**, weaker: **read**)

From the totality of all comparisons the smallest (`bottom`) and the biggest (`top`) action follow.

All nodes are directly represented as Topic Map topics. For theme topics any taxonometric information is directly modeled with the onboard means available for Topic Maps, specifically transitive subclassing and instance-of relationships. In any case we flag theme nodes to be instances of *`theme`*, e.g.

finance isa **theme**

In the same vein, user nodes are marked as instances of *user*:

bill isa **user**

All content nodes are simply instances of *node*. If such a node is affiliated with a theme, this is modeled with an otherwise arbitrary association, for example for *budget_2008*:

dc:subject (node: **budget_2008**, theme: **finance**)

Hereby we made use of the predefined `subject` property inside Dublin Core vocabulary.

Whenever a user is granted a privilege relative to the theme, this will also be represented natively in a map:

privilege (theme: **finance**, user: **bill**, action: **edit**)

Requests for privileges look similar, except that associations representing them are scoped as *pending*:

privilege @ pending (theme: finance, ...)

5. Conclusions

The formalized version of the AAF is the end result of a series of prototype implementations using a scripting language together with one of the mainstream wiki software (Perl + TWiki). In hindsight, that particular platform has not proven to be flexible enough for two necessary adaptions to an existing system:

- the implantation of an ontological backbone, in which to host taxonometric and other semantic network information, and

- the injection of an authorization layer to implement AAF's functionality.

The main motivation for formalization lied in the perspective to better analyze security implications and to describe possible attack vectors. It also opens a foundation to formulate statistical means to subvert an AAF-operated system.

At this stage we have little operational experience with an AAF deployment, not only because of the unsuitability of the chosen platform, but also mostly because of a lack of a mature Topic Maps implementation which allows to quickly scale to hundreds of users and thousands of topics.

This shortcoming had been addressed recently, so that a reimplementation with a CMS but also a conceptual integration with content frameworks such as JSR-283[11] can be attempted.

To substantiate our claim that an AAF-driven system will cause and sustain an adequate and balanced privilege distribution, our efforts will have to concentrate on developing metrics to measure this balance. It is yet unclear whether such metrics will depend on the social setting, be it a corporate environment, a group of cooperating NGOs or independent individuals.

6. Notes and References

[1] Barbera, Michele; Di Donato, Francesca. Weaving the Web of Science. HyperJournal and the Impact of the Semantic Web on Scientific Publishing , Proceedings of the 10th International Conference on Electronic Publishing, Bansko, Bulgaria, 14-16 June 2006.

[2] Annotation and Navigation in Semantic Wikis, Eyal Oren, Renaud Delbru, Knud Moeller, Max Voelkel, and Siegfried Handschuh, Proceedings of the First Workshop on Semantic Wikis, 2006, Ed. Max Voelkel

[3] Costa Oliveira, Edgard; Lima-Marques, Mamede. An Architecture of Authoring Environments for the Semantic Web, Proceedings of the 10th International Conference on Electronic Publishing, Bansko, Bulgaria, 14-16 June 2006.

[4] Resource Description Framework (RDF) model and syntax specification, Technical report, W3C; O. Lassila and K. Swick

[5] TMDM, ISO 13250-2: Topic Maps - Data Model, Lars Marius Garshol and Graham Moore, 2003-11-02

[6] Knowledge-Oriented Middleware Using Topic Maps, Robert Barta, TMRA 2007, Leipzig, (to appear in TMRA 2007 Proceedings, Springer LNCS/LNAI)

[7] Pepper, S. and Moore G.: XML Topic Maps (XTM) 1.0. TopicMaps.Org http://www.topicmaps.org/xtm/1.0/, 2001

[8] Sowa J, et. al. *Knowledge Representation: Logical, Philosophical and Computational Foundations*, Brooks-Cole, Pacific Grove 2000

[9] International Organization for Standardisation, ISO/IEC 13250, Information technology – SGML Appliations – Topic Maps, ISO, Geneva 2000

[10] Pepper S., The TAO of Topic Maps – Finding the Way in the Age of Infoglut, Ontopia http://www.ontopia.net/topicmaps/materials/tao.html, April 2002

[11] JSR 283: Content Repository for Java Technology API Version 2.0, http://jcp.org/en/jsr/detail?id=283