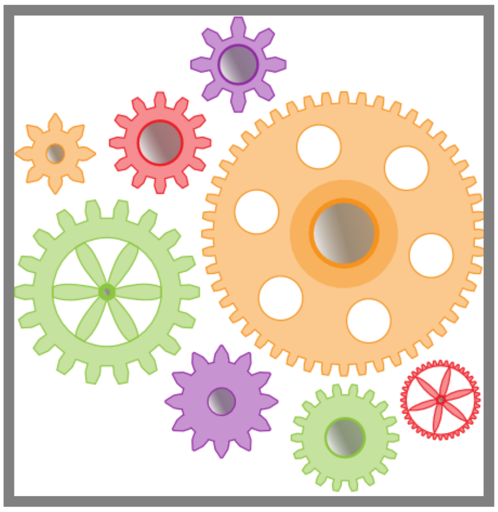
Semantic Technology Access Control Using Datasets

by Mike Bergman - Monday, February 18, 2013

http://www.mkbergman.com/1622/semantic-technology-access-control-using-datasets/



Part 6 in the Enterprise-scale Semantic Systems Series

The fulcrum by which semantic technologies work within the enterprise is the ***dataset***. A dataset refers to a named grouping of records, best designed as similar in record types and intended access rights (though technically a dataset is any named grouping of records).

Datasets play a central role in the organization of information in Structured Dynamics' (SD) open semantic framework (OSF). Datasets are one of the three major access dimensions to the OSF (the other two being ***users/groups*** and ***tools/endpoints***). In combination, these three dimensions -- datasets, users/groups and tools/endpoints -- can also result in a powerful set of profiles that govern overall access to content.

Specific security aspects of the semantic enterprise stack are discussed in another part of this Enterprise-scale Semantic Systems (ESSS) series, but the interplay of those aspects with datasets is fundamental. As such, how datasets are bounded and organized (and, then, named) is a critical management consideration for enterprises that adopt a semantic technology stack based on an architecture like OSF. This role of datasets, how to organize them, how to manage them, and also some best practices for how to use them, are the focus of this part in our series.

Access Dimensions to the OSF

To briefly recall the architectural discussion in this series, SD's semantic technology stack involves a Web services layer (structWSF) used to access specific functional endpoints, all via HTTP queries [1]. Some of these endpoints access complete applications in such areas as tagging, imports/exports, search and the like. Other endpoints individually provide (or not) access to CRUD (*create - read- update -*

delete) rights to interact with either individual records, full datasets, or the ontologies that are the "schema" overlying this information. The net result, at present, is more than 20 individual Web service endpoints to query and interact with the system:

Web Service	Create	Read	Update	Delete
Auth Registrar: Access	X	X		
Auth Registrar: WS	X	X		
Auth: Lister		X		
Auth: Validator		X		
Ontology: Create	X			
Ontology: Read		X		
Ontology: Update			X	
Ontology: Delete				X
Dataset: Create	X			
Dataset: Read		X		
Dataset: Update			X	
Dataset: Delete				X
CRUD: Create	X			
CRUD: Read		X		
CRUD: Update			X	
CRUD: Delete				X

Search		**X**
SPARQL		**X**
Tracker: Create	**X**	
Scones		

This structWSF Web services layer has a three-dimensional design that is used to govern access:

1. Users (or Groups or Roles)
2. Tools, and
3. Datasets.

A "user" may extend from an individual to an entire class or group of users, say, unregistered visitors to a given portal. Tools refer to each of the structWSF endpoints, each with its own URI.

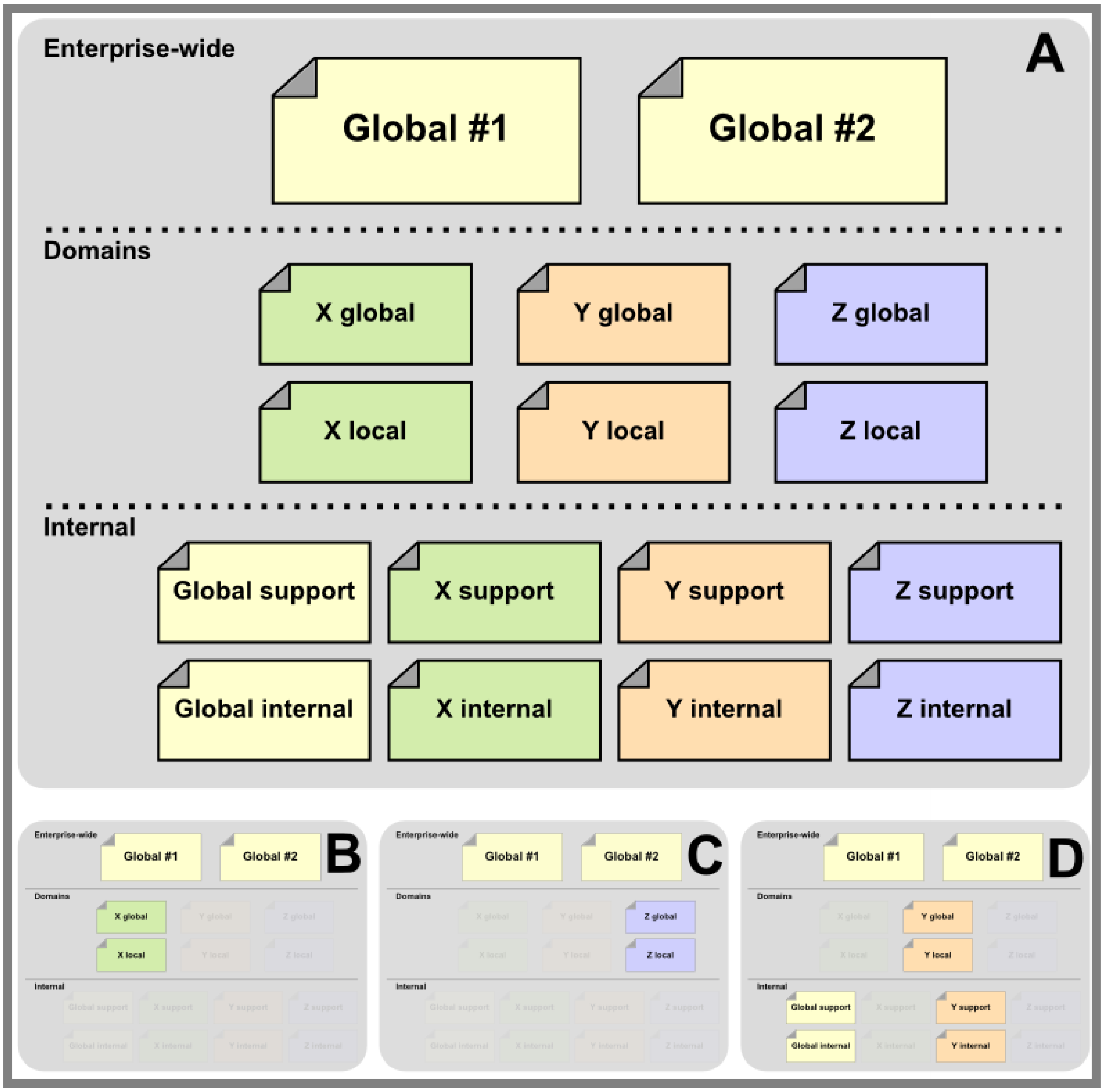
What this means is that a given user may be granted access or not -- and various rights or not from reading to the creation or deletion of information -- in relation to specific datasets. Stated another way, it is in the nexus of user type and dataset that access control is established for the semantic system.

In an enterprise context, a given individual ("user") may have different access rights depending on circumstance. A worker in a department may be able to see and do different things for departmental information than for enterprise information. A manager may be able to view budget information that is not readable by support personnel. A visitor to a different Web site or portal may see different information than visitors to other Web sites. Supervisors might be able to see and modify salary data for certain employees that is not viewable by others.

The user role or persona thus becomes the access identifier to the system. What information and what tools they might use in relation to that information is defined in relation to the datasets for which they have access.

Some Access Scenarios

So, let's say, a given enterprise has two major information stores, #1 and #2, and also has some domain (or departmental or other such boundary) information for X, Y and Z, some of which is local (perhaps for the local branch) and the rest global (for that line of business). Further, let's also suppose that those same departments also have sensitive, internal information related to either internal matters (such as salaries) or support matters (such as qualified vendors). This basic scenario is laid out in **A** of the diagram below:



Now, depending, different individuals (most often assigned to different access groups, but that is not required) need to have different access to this information. In one case, a general user with access to mostly public stuff exists for domain **B**; another for domain **C**. Then still, a supervisor or someone internally may have responsibilities in the Y domain; that could be case **D**.

Any of the same variations above could result in a different use case; **A** - **D** above is merely illustrative.

Profiles to Overcome the Combinatorial Problem

It is fairly easy to see that the combination of datasets **x** tools **x** roles can lead to many access permutations. With, say, the current 20 some-odd tools in the OSF with five different roles and just ten different datasets, we already have about 1,000 permutations. As portals and dataset numbers grow, this combinatorial explosion gets even worse. Of course, not all combinations of datasets, tools and roles make sense. In fact, only a relatively few number of patterns likely covers 95% or more of all likely access options.

Because access rights are highly patterned, these theoretical combinations can in fact be boiled down to a small number of practical templates — called ***profiles*** — to which a newly registered dataset can be assigned. (Of course, the enterprise could also tweak any of the standard profiles to meet any of the combinatorial options for a specific, unusual individual, such as for a tax auditor.) Experience, in fact, shows the number of actual profiles to be surprisingly small.

For instance, consider these possible profile patterns:

- **Profile: Public** (standard) — this profile is for a dataset intended for broad public access
- **Profile: Registered** — this profile is for datasets that are limited to registered users of a ***portal*** (possibly as a way to prevent spam or to encourage membership or participation)
- **Profile: Curated** — this profile is where a specific group or groups (which themselves can be flexibly determined and assigned) has curation rights for the dataset, or
- **Profile: Internal** — this profile is for internal (private) datasets where only a specific group or groups may access or modify. In some instances, an internal dataset might be the profile type while the dataset is under development, with the profile shifting to a broader access category once completed.

Profiles may, of course, be applied to any permutation.

This profile concept can now be expanded to incorporate user type. Four categories of users can illustrate this dimension:

- **O** = Owner (the original registrar of the dataset; often possibly the “owner” or "admin" of the ***portal***, but not necessarily so)
- **G** = Group member (a registered user who is a member of a specific group)
- **R** = Registered user (an authorized ***portal*** user with a Drupal login and password)
- **P** = Public (anonymous user)

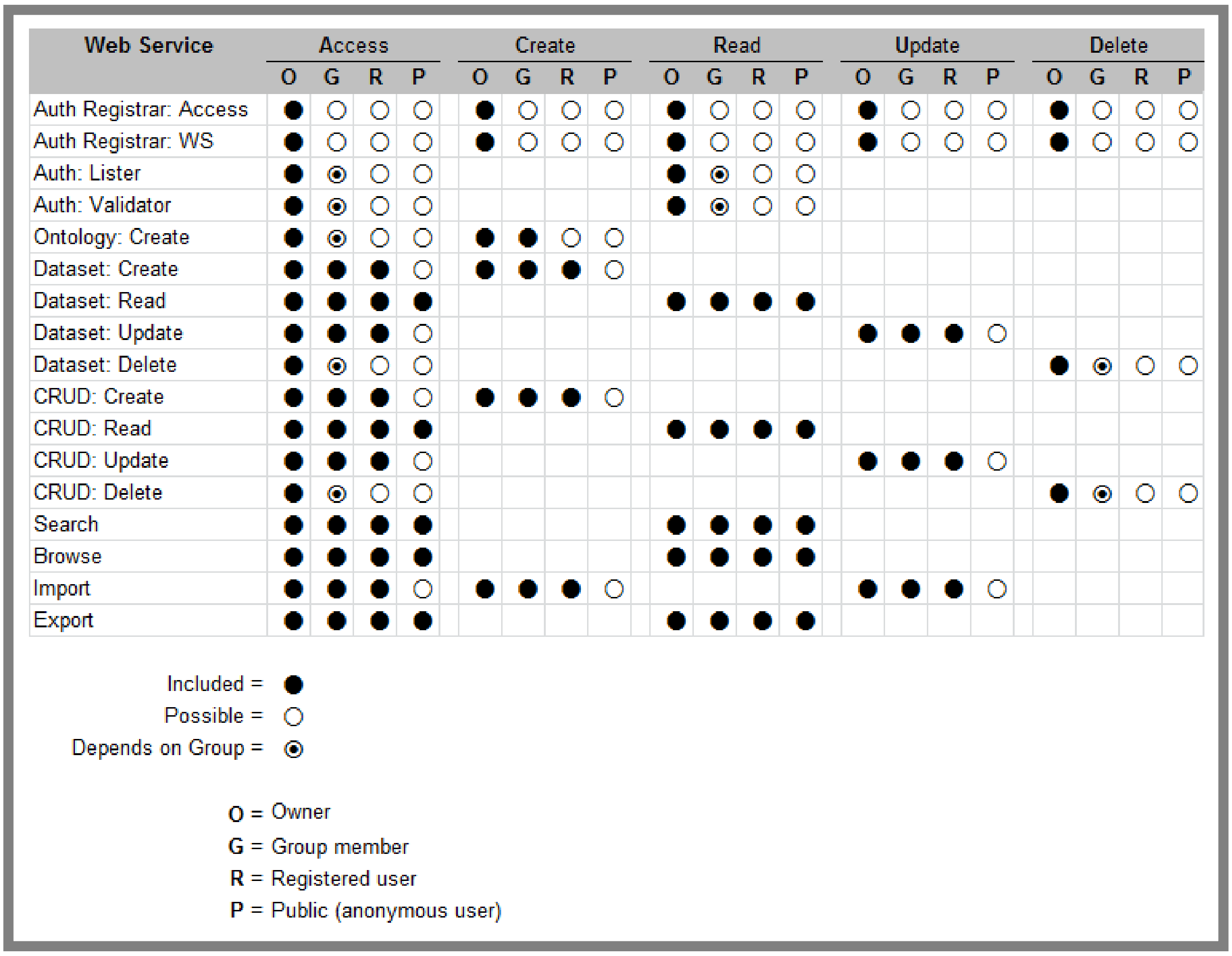
Further, of course, with a multitude of groups, there are potentially many more than four categories ("roles") of users as well.

A Sample Profile Matrix

To illustrate how we can collapse this combinatorial space into something more manageable, let’s look at what one of the profile cases noted above — that is the **Public** profile — can now be expressed as a pattern or template. In this example, the **Public** profile means that owners and some groups may curate the data, but everyone can see and access the data. Also note that export is a special case, which could warrant a sub-profile.

We also need to relate this **Public** profile to a specific dataset. For this dataset, we can characterize our "possible" assignments as described above as to whether a specific user category (**O**, **G**, **R** and **P** as noted above) has available a given function (open dot), gets permission rights to that function by virtue of the assigned profile (solid dot), or whether that function may also be limited to a specific group or groups (half-filled dot) or not.

Thus, we can now see this example profile matrix for the **Public** profile for an example dataset with respect to the available structWSF Web services:



Web Service	Access				Create				Read				Update				Delete			
	O	G	R	P	O	G	R	P	O	G	R	P	O	G	R	P	O	G	R	P
Auth Registrar: Access	●	○	○	○	●	○	○	○	●	○	○	○	●	○	○	○	●	○	○	○
Auth Registrar: WS	●	○	○	○	●	○	○	○	●	○	○	○	●	○	○	○	●	○	○	○
Auth: Lister	●	⦿	○	○					●	⦿	○	○								
Auth: Validator	●	⦿	○	○					●	⦿	○	○								
Ontology: Create	●	⦿	○	○	●	●	○	○												
Dataset: Create	●	●	●	○	●	●	●	○												
Dataset: Read	●	●	●	●					●	●	●	●								
Dataset: Update	●	●	●	○									●	●	●	○				
Dataset: Delete	●	⦿	○	○													●	⦿	○	○
CRUD: Create	●	●	●	○	●	●	●	○												
CRUD: Read	●	●	●	●					●	●	●	●								
CRUD: Update	●	●	●	○									●	●	●	○				
CRUD: Delete	●	⦿	○	○													●	⦿	○	○
Search	●	●	●	●					●	●	●	●								
Browse	●	●	●	●					●	●	●	●								
Import	●	●	●	○	●	●	●	○					●	●	●	○				
Export	●	●	●	●					●	●	●	●								

Included = ●
Possible = ○
Depends on Group = ⦿

O = Owner
G = Group member
R = Registered user
P = Public (anonymous user)

Note, of course, that these options and categories and assignments are purely arbitrary for our illustrative discussion. Actual needs and circumstances may vary wildly from this example.

Matrices such as this seem complex, but that is why profiles can collapse and simplify the potential assignments into a manageable number of discrete options. If the pre-packaged profiles need to be tweaked or adjusted for a particular circumstance, provisions through the CMS enables all assignments to be accessed in individual detail. Via this design, knowledge and collaboration networks can be deployed that support an unlimited number of configurations and options, all in a scalable, Web-accessible manner. The data that is accessed is automatically expressed as linked data. This same framework can be layered over *in situ* existing data assets to provide data federation and interoperable functionality, all responsive to standard enterprise concerns regarding data access, rights and permissions.

Best Practices

Datasets are clearly one of the fundamental dimensions for organizing content within this OSF semantic enterprise design. Some of the best practices in bounding these structures:

- **Domain** - what is the applicable scope or business purpose of this information? It is best to think of this question with regard to access, which is, after all, the most pragmatic way to think of it
- **Source** - does the data vary by publisher or source location? For example, provenance or download location or format may be an important distinguishing factor in release or access, and may have copyright or royalty implications
- **When created** - does the data have periodic update or creation times? For example, it may be important to distinguish between preliminary data and final data or to segregate data because of workflow or processing considerations
- **Access rights** - are there any differences in how users may see or act upon the data? For example, privileged budget information may be put in a different dataset from public financial information
- **Type** - does the data vary by class or kind? For example, records about schools might be desirable to keep different from records about churches, though at a different level both may be considered buildings, or
- **Attributes** - are there differences in fields or attributes that describe the data? For example, a portion of records may have complete attribute descriptions, while the majority only contain a few descriptive fields.

Any of these differences may warrant creating a separate dataset. There are no limits to the number of datasets that may be managed by a given OSF instance.

Once such boundaries get set, then thinking about common attributes or metadata should be applied. Still further, datasets and their records (as all decision or information artifacts in an enterprise) go through natural work stages or progressions. Even the lowliest written document needs to be drafted, reviewed, characterized, approved, and then possibly revised. Whatever such workflow steps may be, including versioning, may warrant consideration as belonging to a different dataset.

Lastly, whatever the operational mode devised, finding naming conventions to reflect these variations is essential to manage the dataset files. Which goes to show: datasets are meaningful information artifacts in and of themselves.

NOTE: This is part of an ongoing series on enterprise-scale semantic systems (ESSS), which has its own category on this blog. Simply click on that category link to see other articles in this series.

[1] Or programmatically via the structWSF API.