Domain-specific Instantiations Based on the *Open Semantic Framework*

by Mike Bergman - Thursday, June 17, 2010

http://www.mkbergman.com/891/domain-specific-instantiations-based-on-the-open-semantic-framework/

Structured Dynamics Completes Design Phase; Citizen Dan First Exemplar

Structured Dynamics has been in a fervent -- and, we believe, fruitful -- design phase for the past 18 months. All of the working parts related to how to embrace becoming a semantic enterprise have now been defined and designed. Actual tools and components accompany many of these parts and have been deployed.

Recently, I have been speaking and blogging much about rationale, process, mindset and approach for how to bring semantics into the organization. But, prior to now, we have not spoken much about the overall *design* behind our approach. Today, as we complete our design phase and introduce our first exemplar instance of it -- Citizen Dan [1] -- we are finally in a position to describe this overall approach.

We term our approach the ***open semantic framework***, also ***OSF***. The open semantic framework is a combination of a layered architecture and modular software. The open semantic framework represents the **software** component of the four-component *total open solution*, recently described in a three part series. I return to this topic in the conclusion of this post.

Revisiting Design Objectives

Over the past nine months, I have been focusing my writing largely on the semantic enterprise, with more specificity regarding our Open SEAS (*Semantic Enterprise Adoption and Solutions*) initiative. In bits and pieces, these writings have tended to reflect a number of objectives:

- Leverage existing information assets (data + structure) as much as possible
- Develop incrementally, and validate and justify as you go
- Emphasize, where possible, open standards and open software
- Employ Web-oriented architectures
- Adopt an open-world approach that acknowledges that information is most often incomplete; the approach is a key enabler for incremental deployments
- Use URIs as object identifiers, and use linked data where practical
- Embrace any data format found in the wild, but use RDF as the ultimate integration data model
- Design architectures and APIs that avoid "lock-in" and support multiple tools options across the

stack
- Provide systems and capabilities that put all information sources -- text, media, semi-structured and conventional databases -- on an equal footing
- Promote designs that bring the ability to create useful results into the hands of users and decisionmakers; relegate IT to a support role.

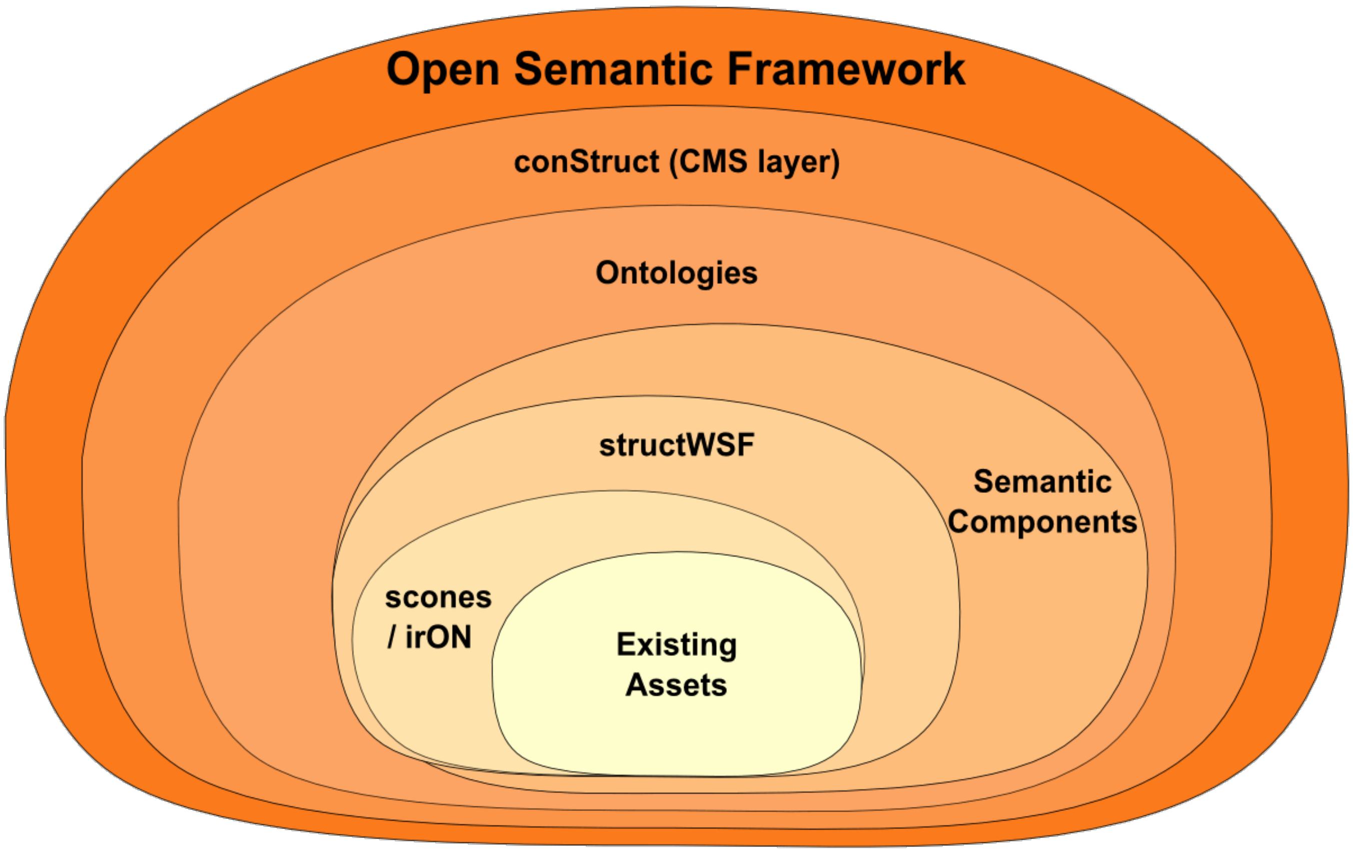
To date, the result of these design objectives is perhaps best captured in my Seven Pillars of the Open Semantic Enterprise posting, as well as our general discussions regarding adaptive ontologies. Yet, still, these writings have been somewhat piecemeal. What this document attempts to do is to place all of these perspectives into a single, coherent whole.

The Incremental Layers of the Open Semantic Framework

Structured Dynamics has been a strong advocate for layered architectures, with clear APIs between layers as appropriate. But these layers are not "laminates" that completely cover the layer below, nor are they all needed or necessary. Depending on the circumstance, some layers are unneeded or superfluous. Layers may be added or not incrementally.

In this manner, then, the open semantic framework is perhaps more akin to a pearl, than to a laminate or cocoon. Each subsequent layer does not "embed" the layer prior to it, and some layers actually may inter-operate with multiple layers below or above it (this is notably true for the "ontologies" layer, which has interactions up and down the stack).

Nonetheless, we can envision this pearl of the open semantic framework and its layers as follows:



(click for full size)

Others have termed this the "semantic muffin" or even "semantic muppet" or "semantic blob". Whatever (hehe). The real idea is that layers may accrete (as in the growth of a pearl) and occur over time and be uneven. Each layer, though, does have a role to play (though it may not be needed in a given deployment), and does act to augment existing information assets in the transition to a semantic framework. Beginning at the core, each of these layers -- with external references as appropriate for more details -- is described below.

Existing Assets Layer

The open semantic framework is premised on leveraging existing information assets. Sure, once the framework is in place, new information can be brought into it in a more direct, semantic manner. But, the real thrust and benefit of this framework is to provide an incremental pathway for finally inter-operating and federating prior decades of data, structure and information assets.

These information assets may reside inside or outside the enterprise. They may (and DO!) exist in many formats and are described by many schema. They may come from internal transaction systems or warehouses, or may exist external on the Web or at supplier or partner sites. These information assets may span from conventional databases and relational data systems to XML interchange standards, Web pages and standard internal text or documents. In short, there is NO information asset that is not amenable to be included in this framework.

The Information Transformation (scones/irON) Layer

The information transformation layer provides either: 1) extraction of concepts and entities as structured metadata from source text or documents; or 2) conversion of existing data assets to interoperable form. As implemented by Structured Dynamics, the extractions are conducted by either scones (*Subject Concept or Named EntitieS*) or third-party utilities, and the conversions occur via irON (*instance record Object Notation*) or third-party "RDFizers".

Depending on the source, the net result of the transformation is to produce interoperable data and information that can be ingested and used by other layers in the framework.

Though not strictly analogous, this layer bears some resemblance to the ETL (extract, transfer, load) utilities used in many enterprise information integration applications. Unlike those conventional systems, this information transformation layer also may capture and represent some of the source schema.

In all cases, however, these transformations are relatively simple and get parsed against the available structure (the ontologies, schema and entity reference lists) in the system to generate the semantic metadata (tags).

At this point, the extracted structure is generally at the level of instance records, or the ABox, with simple assertions of attribute-value pairs for specific records [2]. Little schema transformation or mapping occurs at this layer (if such is needed, that occurs at the structWSF layer; see next). Actual federation or interoperation occurs at later layers based on the TBox structures [2].

This modular portion of the framework is explicitly designed with APIs to allow third-party tools to be plugged in and substituted.

The structWSF Layer

The major workhorse of the open semantic framework is the structWSF (Web services framework) layer. structWSF is the most complicated of the OSF layers and has many supporting software packages and capabilities. The structWSF layer provides the standard, common interface ("canonical") layer by which existing information assets get represented and presented to the outside world and to other layers in the OSF stack.

structWSF is a platform-independent Web services framework for accessing and exposing structured RDF data. Its central organizing perspective is that of the *dataset*. These datasets contain instance records, with the structural relationships amongst the data and their attributes and concepts defined via ontologies (schema with accompanying vocabularies; see below).

The structWSF middleware framework is generally RESTful in design and is based on HTTP and Web protocols and open standards. The current structWSF framework comes packaged with a baseline set of about twenty Web services in CRUD, browse, search and export and import. All Web services are exposed via APIs and SPARQL endpoints. Each request to an individual Web service returns an HTTP status and optionally a document of *resultsets*. Each results document can be serialized in many ways, and may be expressed as either RDF or pure XML. An internal representation, structXML [3], is used for internal communications across all structWSF Web services and with other layers.

structWSF has a central service that governs access rights and permissions. These rights occur at the level of the dataset, which gives immense flexibility to how data may be accessed, read, modified, created or deleted (or not). Datasets within a given structWSF instance may be accessed directly via API or via SPARQL queries to the instance's endpoint. Depending on rights and query, results sets may be returned from a given structWSF instance in an infinite variety of ways.

This latter capability is the essential interface for subsequent layers in the open semantic framework stack. Depending on those subsequent components, pre-staged data and results sets may be returned for an essentially limitless variety of purposes.

Each structWSF instance also has a unique Web address that enables one or a multitude of instances to communicate and share with one another. This simple, but elegant, method enables structWSF instances to participate or not in potentially global or restricted local networks and collaboration environments. This is currently the largest untapped potential of structWSF with respect to its existing deployments.

The Semantic Components Layer

The newest layer in the stack is the semantic components layer. This layer takes results sets -- most often generated by a specific query or data slice request -- from one or more structWSF instances and then presents that information via a variety of data visualization or data presentation widgets (what we specifically call '*semantic components*' due to their design [4]). The operation and sensitivity of these display components are themselves driven by a presentation and data analysis (including statistics) ontology.

Current display widgets include: filter; tabular templates (similar to infoboxes); maps; bar, pie or linear charts; relationship (concept) browser; story and text annotator and viewer; workbench for creating structured views; and dashboard for presenting pre-defined views and component arrangements. These are generic tools that respond to the structures and data fed to them, adaptable without modification to any domain.

As presently implemented by Structured Dynamics, this layer consists either of Flex data visualization components or structured data display templates based on Smarty. The inherent design allows for updates to other bases (such as HTML5). The layer may also be swapped out or substituted with third-party capabilities.

The strength and power of this system is governed by its own ontology, the Semantic Component Ontology (SCO) (see next).

This is an extremely flexible layer in the open semantic framework stack. Expect an ongoing series of explanatory blog posts and online resources in the upcoming weeks to explain this innovative capability.

The Ontologies Layer

The ontologies layer actually refers to all structured assets driving the system. As such, this layer might be considered the "brain" (though rather simply specified!) of the open semantic framework.

At a true schema or TBox level [2], the ontologies layer represents the concept and relationships of the domain at hand. This layer also hosts the specific local entities and prominent things (people, places, events, etc.) useful for extracting local and domain-specific relevance. However, those views are also supplemented with some administrative ontologies (two examples are SCO and irON) that guide how the user interfaces or widgets in the system should behave.

The concept level represents the "world view" of the specific instantiation of the open semantic framework at hand. This conceptual (TBox) view provides the structural organization of information, inferencing capabilities, and navigation, faceting and explorer structure. The entity (ABox) view provides tagging for prominent individuals and instances important to the domain at hand, and guides the structure behind data visualizations of attribute or indicator data.

The administrative level uses simple roles and relationships for attributes and indicators to inform the framework as to how and with what widget to display information. For example, a "type" of information that is geographically related can be instructed to use the map component as an option for display. Whether some information is used for totals, comparison purposes, or other specifications useful to data visualization and graphing may also be specified.

The language and relationships (predicates or properties) of these administrative ontologies are simple and straightforward. It is, for example, relatively easy to define data display functions at the broad dataset and attributes level. Simple determinations drive how results sets and their associated results types may be displayed, no matter what datasets or slices may be generated as a result of the queries or requests fed to the system.

The structure in these layers can be replaced by other structures for other instantiations and circumstances. Indeed, all other layers in the open semantic framework can remain relatively fixed while tailoring the instance to new domains solely via this layer. The ontologies layer is what gives any given instantiation of OSF -- such as Citizen Dan -- its unique focus and scope.

The Content Management System (conStruct) Layer

The thinnest layer (that is, least substantial with respect to this framework) is the content management system (CMS) layer. In its current form, the open semantic framework uses the Drupal CMS via our conStruct plug-in modules. The design of the framework, however, has explicitly accommodated the possibility that other CMSs may substitute for this role.

The CMS layer is optional if structWSF endpoints are sufficient or if simple Web pages hosting semantic components are deemed as adequate. Very small organizations or deployments may reasonably choose to have no CMS layer at all.

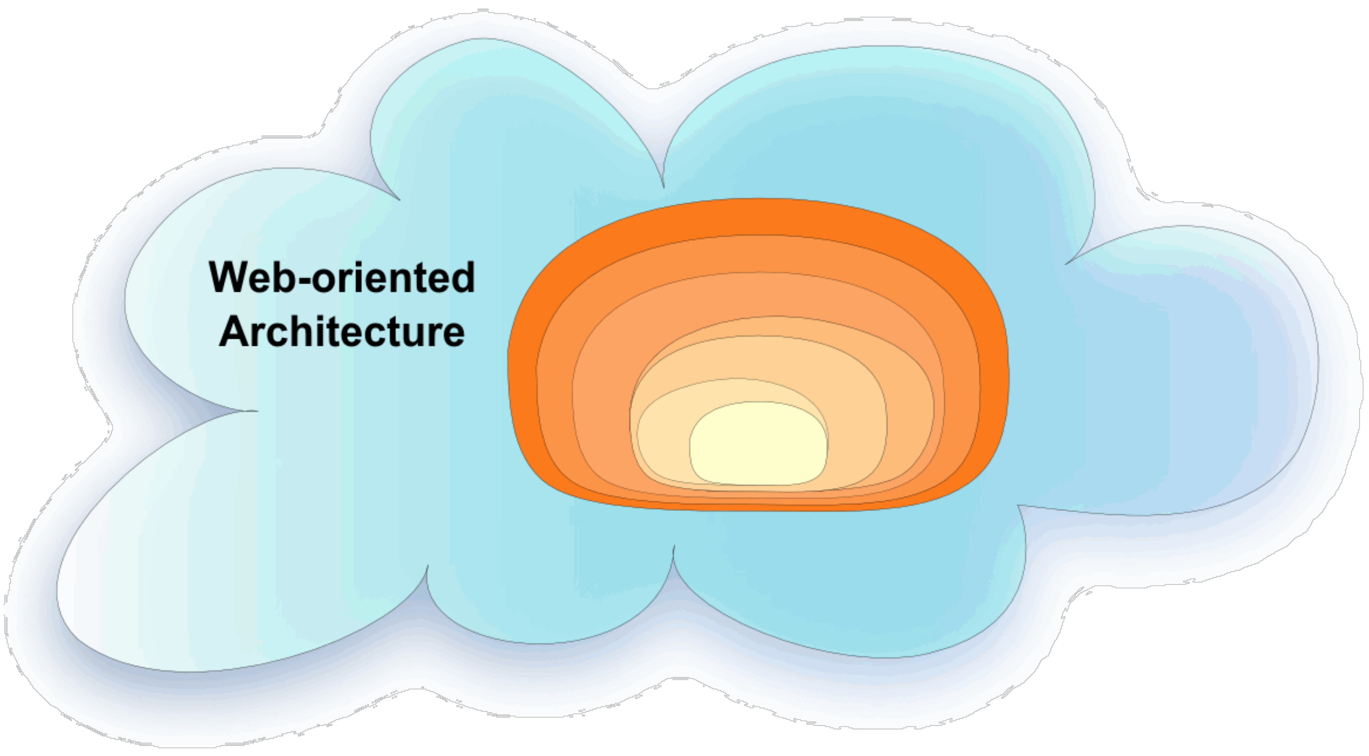
However, for most sites or portals with more than a few active users, it is desirable to have broad flexibility in theming ("skinning"), user rights and permissions, or other functionality. These are the roles of the CMS layer. Drupal, for example, is presently supported by more than 4500 third-party modules in every conceivable function, from polling to blogs and rating systems and bulletin boards.

For such generalized portals or collaboration environments, it makes sense to adopt and install a flexible CMS system, such as Drupal. Much of the user experience and functional environment can be provided through such means.

The open semantic framework is thus designed to reside easily in a CMS while also providing the hooks to take advantage of the generalized user rights and functionality of the CMS. In this manner, the open semantic framework is able to stay focused on its structured data and interoperability purposes, while still gaining the advantages of rich-featured content management systems.

The OSF is a Web-oriented Architecture

With its inherent open-world orientation [5] and distributed and collaborative potential, the open semantic framework was designed from the outset to be Web-capable and Web-oriented:



(click for full size)

A Web-oriented architecture (WOA) has a number of understood requirements, to which the open semantic framework adheres. Specifically, these design considerations support the framework as being part of WOA:

- Data and objects are all identified with Web addresses (URIs)
- Data is generally exposed (and universally available) as linked data
- SPARQL endpoints and APIs are generally RESTful in design
- The overall architecture is modular, with inherent decentralized and distributed aspects
- All display and visualization aspects are cross-browser ready and capable.

OSF is the Basis for Domain-specific Instantiations

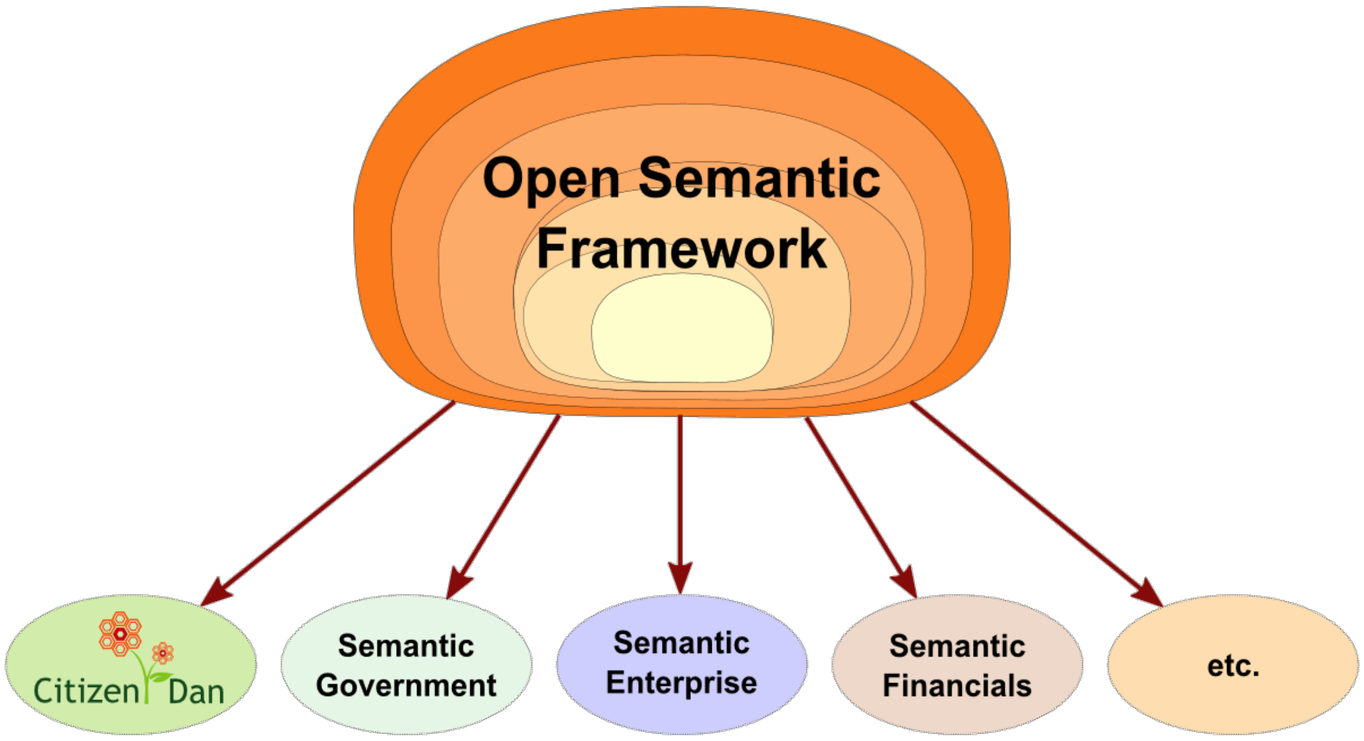
Citizen Dan is our first exemplar instance of this open semantic framework. The details page for the project goes into some of Citizen Dan's functionality and capabilities.

Citizen Dan is specifically geared to local governments and localities, with an emphasis on community indicator systems (CIS). CIS have become a popular way of measuring and tracking measures of local economic and social well-being; they are closely related to sustainability and how to measure it as used in many economic and environmental domains.

However, in the context of this post, what is really interesting about Citizen Dan is that its semantic framework is a completely open and generic one. The same set of tools and capabilities described on its details page can be applied to any domain that needs to manage and understand information in its own domain. This includes from unstructured text or documents to conventional structured databases.

What changes from domain to domain are the data structures (the ontologies, schema and entity reference lists; see above) that are fed to this open semantic framework. By swapping out new structures, what can be called *Citizen Dan* in one instance can morph to become *Curriculum Carla* in say, the education instance or *Doctor Doolittle* in the veterinary science instance [6].

We can illustrate these multiple instances as follows:



(click for full size)

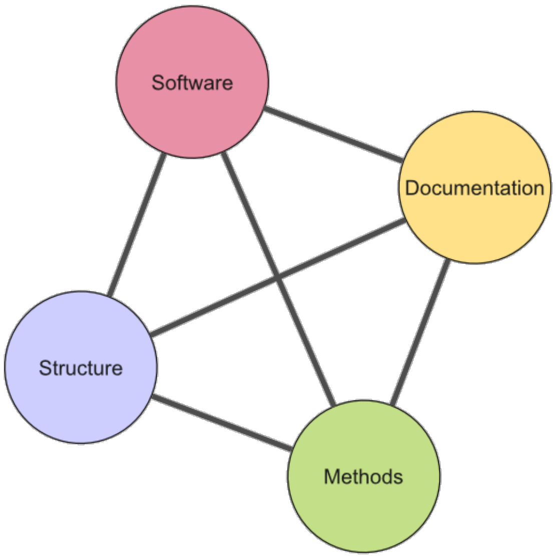
What this figure illustrates is that even a branded expression of the framework -- such as Citizen Dan -- is merely an instance of that framework. And, actually, when expressed in such a packaged manner, we can more accurately call the standard and bundled suite of generic functions and accompanying structure of Citizen Dan as an instantiation of the open semantic framework:

in·stan·ti·ate\in-**?**stan(t)-sh?-?t\ *(transitive verb)* is to:

1. (transitive) to represent an abstract concept by a concrete instance
2. (transitive, object orientated computing) to create an object (an instance) of a specific class

in·stan·ti·a·tion\in-'stan(t)-sh?-?-sh?n\ *(noun)* [7]

By replacing the structure bases, and by tailoring the function suite appropriate to a given market and use, we can create many instantiations of the open semantic framework for different domains and markets. In this manner, Citizen Dan can be seen as an early exemplar of the framework, but not as a definer and limiter to it.



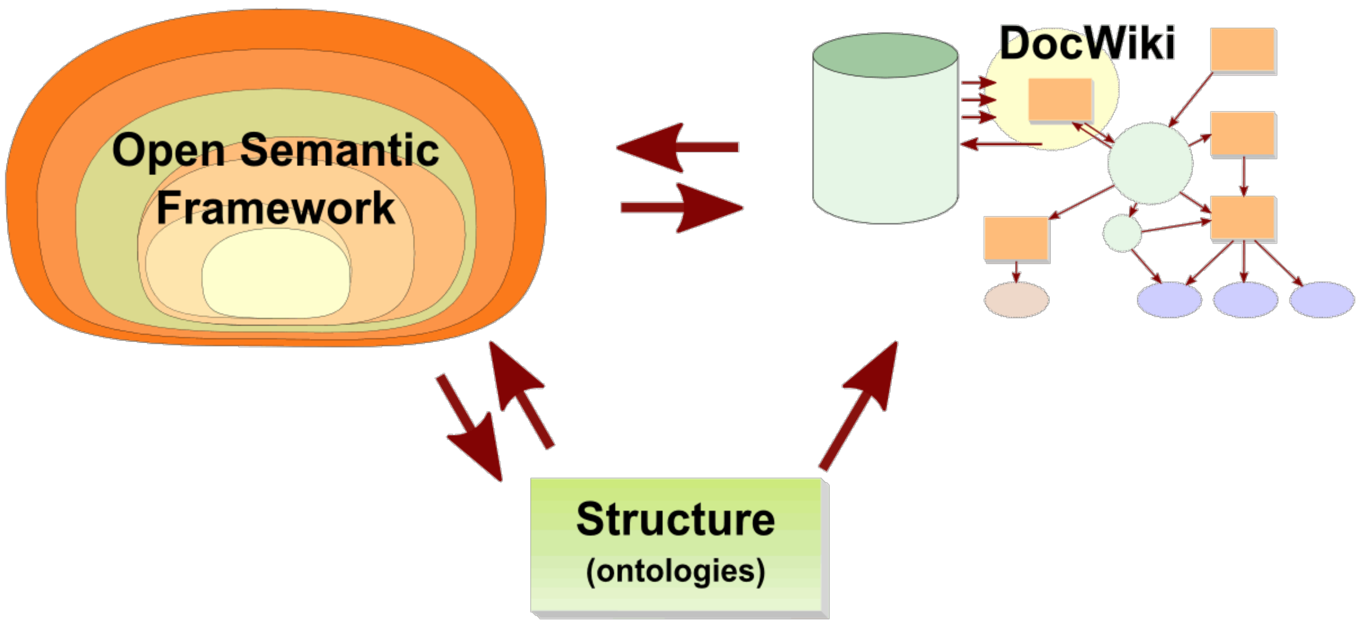
OSF is the Software Leg to a '*Total Open Solution*'

So far, this discussion has focused solely on considerations of software and architecture. While we see the power of the open semantic framework, highly useful in itself, this is inadequate alone to achieve acceptance and success in the enterprise (as we noted in our most recent posts). The very forces that are compelling enterprises to look at new options, are also the same ones that pose difficult hurdle rates for acceptance of open source.

To address this issue, we have developed a four-legged foundation to what we termed the *total open solution*. The solution involves **software**, **structure**, **documentation** and **methods** (or best practices). Each of these connect and relate to the other foundations.

The open semantic framework is clearly the software (and architecture) leg to this foundation. Again, however, what is interesting is that the mere swapping out of the structure can also make the system relatively ready for other domains.

We see these relationships in the following diagram, that also shows that the DocWiki portions of the solution embody the documentation (aside from code-level comments) and methods legs of the foundation:



(click for full size)

Differences between domains may also lead to differences as to which components are included or not in that domain's desired instantiation.

The hugely important implied point, however, from the diagram above, is to show how nearly universal the content and methods in the DocWiki may be to other domains. Because the deltas between domains largely result from structure and what specific functional components are included or not, it becomes clear that most documentation and practices shared with the DocWiki will be applicable across domains. Sure, the use cases and some of the specific terminology may change, but we can also now see a high degree of re-usability of documentation and knowledge base across markets. This realization makes the usefulness and leverage of the DocWiki even higher.

A Common Language and Framework for Moving Forward

Developing "common language" by which to describe and convey things -- especially new things like semantics that also have strong technical aspects -- is tough, very tough. We are only now beginning on this process; we look to many in the community and elsewhere to help define informative and evocative terminology.

Per the original design objectives above, Structured Dynamics has approached the challenge of the semantic enterprise in what we think is both a pragmatic and a new way. The insistence on preserving and respecting existing information assets, matched with the opportunities and different mindsets arising from an open-world approach [5], have necessitated thinking through new designs and developing new concepts. Any time such new thinking and concepts occurs, new language and new metaphors must accompany it.

While certainly there are components and various software packages that populate and comprise an open semantic framework, the framework is also just as importantly a world view or way to think about information, information development, and its architecture. For example, a pivotal concept is that an open semantic framework is built around generic tools responsive to the information structures fed to them. This realization shifts the locus of emphasis from software development *per se* to creating, managing and adapting data and information structures. While this democratizes the information development process and is more inclusive of all knowledge workers, it also imposes needs for new toolsets and business processes. We are only at the nascent stages of understanding and learning about these differences.

Similarly, a development approach that is inherently incremental and leverages (rather than replaces or displaces) existing information assets means IT projects need to be considered in a new light. Small projects with more emphasis on tangible and demonstrable benefits will alter budgets, lower risks, and place a need for quicker turnaround. Like the architecture of the open semantic framework itself, projects based on OSF are also more distributed, decentralized and modular.

With such decentralization also comes the need for mechanisms and systems to overcome vendor "lock-in" and proprietary systems. A key thrust in support of what we have called the *total open solution* and its mixture of documentation and methods to accompany software and structure is specifically targeted at this issue. Tools and means for collaboration and concurrent contributions are another possible answer. Prior software practices in agile development and version control will see extensions to all manner of information development across the enterprise.

We are proud of our design work and proof-testing with clients over the past 18 months. We believe the open semantic framework and its implications to be a fundamental shift in how organizations need to think about their information development, existing information assets, and IT budgets and processes. We know widescale adoption is not yet at hand -- enterprises are justifiably conservative when it comes to new thinking. But, given global competition and tight pocketbooks, the open semantic framework is a formulation to which enterprises and governments should pay very close attention.

---

[1] Citizen Dan is an open source system for aggregating different indicator data concerning local, community well-being. Information sources may include the Web, real-time feeds, government datasets, municipal government information systems, or crowdsourced data. Information can range from standard structured data to local narratives, including from minutes and reports, contributed stories, blogs or news outlets. The 'raw' input data can come in essentially any format, which is then converted to a standard form with consistent semantics. See current details with screenshots.

[2] Structured Dynamics' best practices approach makes explicit splits between the "ABox" (for instance data) and "TBox" (for ontology schema) in accordance with our working definition for description logics, a fundamental underpinning for how we use RDF:

"Description logics and their semantics traditionally split *concepts* and their relationships from the different treatment of *instances* and their attributes and roles, expressed as fact assertions. The concept split is known as the TBox (for *terminological* knowledge, the basis for *T* in *TBox*) and represents the schema or taxonomy of the domain at hand. The TBox is the structural and intensional component of conceptual relationships. The second split of instances is known as the ABox (for *assertions*, the basis for *A* in *ABox*) and describes the attributes of instances (and individuals), the roles between instances, and other assertions about instances regarding their class membership with the TBox concepts."

[3] A subsequent post will document this rather straightforward XML schema.

[4] Contact Structured Dynamics for a early sneak peek. The Citizen Dan application will be publicly released as an online sandbox and demo by the end of summer 2010.

[5] See M. K. Bergman, 2009. The Open World Assumption: Elephant in the Room, December 21, 2009. The open world assumption (OWA) generally asserts that the lack of a given assertion or fact being available does not imply whether that possible assertion is true or false: it simply is not known. In other words, lack of knowledge does not imply falsity. Anothe way to say is it that everything is permitted until it is prohibited. OWA lends itself to incremental and incomplete approaches to various modeling problems.

[6] Of course, things are always not so simple as this. The CMS layer gives the open semantic framework the ready ability to change themes and layouts ("skins), not to mention the breadth and specifics of what ancillary site functionality might be provided. Moreover, the module basis of the open semantic framework also means that entire clusters of functionality might be dropped from a given instantiation (or added to it!) without violating or negating this framework.

[7] Dictionary references are from Merriam-Webster and Wikitionary.
