FEA Reference Model Ontologies (FEA RMO)

Version 1.1

February, 2005

Dean Allemang, TopQuadrant, Inc.

Ralph Hodgson, TopQuadrant, Inc.

Irene Polikoff, TopQuadrant, Inc.

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

1. Overview

This white paper describes the design of and the deployment options for the Federal Enterprise Architecture Reference Ontology Models (FEA RMO). The work of representing the FEA as formal ontologies was funded in part by GSA.[1]

This white paper outlines the work to date towards formalizing the FEA RMO. Five models have been encoded in OWL (W3C standard Web Ontology Language). The models have been built with the following three design points in mind:

- Using FEA RMO as a way to provide "line of sight" support, drawing on all the models,
- Using FEA RMO as a way to merge and query FEA related data (i.e., agencies' exhibit 300 submissions and agencies' enterprise architectures), and
- Using FEA RMO as a way to provide an assessment of the FEA models.

The paper is organized in the following sections:

- ***FEA RMO Ontology Models*** – this section describes FEA RMO architecture and identifies business value and potential of the models
- ***Assessment of the Semantics of the FEA*** – this section describes some inconsistencies, conflicts and omissions discovered in the process of formalizing FEA framework as ontology models
- ***Use Cases of the FEA RMO*** – this section describes representative use cases
- ***FEA RMO Development Approach*** – this section describes the design patterns used by FEA RMO; it is intended for people interested in using and extending the models
- ***Tooling Issues*** – this section describes technical issues with the state of the art ontology tools discovered during FEA RMO development
- ***Recommendations and Future Plans*** – this section describes our "wish list", defining where and how we would like to see the work progressing; it also identifies FEA RMO deployment and governance options.

The FEA RMO project has provided an opportunity for us to rigorously study the FEA models. We have found much richness in the specifications. In many places the models are well thought out.

Establishing ontological models is always a balance between expressivity and parsimony and we feel that we achieved good compliance with the guiding principles that were set for us by GSA. The work has been a substantial effort and not without its battles with the tooling currently available in the market-place. Nonetheless the ontologies for PRM, BRM, SRM, TRM and DRM now exist and we look forward to its integration with other tools as well as review and adoption by a wider audience of interested parties.

[1] We would like to thank Rick Murphy, Enterprise Architect and George Thomas, Chief Enterprise Architect of the Office of CIO of GSA for their vision, support and contribution to the use cases described in this white paper.

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

2. FEA RMO Ontology Models

The FEA reference models are designed to facilitate cross-agency analysis and the identification of duplicative investments, gaps, and opportunities for collaboration within and across Federal Agencies. The method for achieving these benefits is to provide a series of five reference models, relating to Performance, Business, Services, Technology and Data.

Reference models provide a means by which enterprise architectures can be designed in a uniform way. If an agency "aligns" its architecture to a reference model, then (at very least) enterprise architects in another agency can understand the components of that agency's architecture, and can understand where there might be possibilities for collaboration or asset re-use.

FEA framework has been developed and documented by the Office of Management and Budget (OMB). OMB made the models available as Adobe PDF documents downloadable from the FEA Program Management Office web site.

This work has encoded FEA models in the W3C[2] standard language for representing semantic models on the web. This language is called OWL (Web Ontology Language). The result is what we call FEA Reference Model Ontology, or FEA RMO.

2.1 Why Formal Reference Models?

Reference models are typically written in natural language, and are presented as some form of human-readable document. The reference models of the FEA are no exception.[3] This form of presentation has the advantage that the reference models can be read by anyone who can read PDF files; but it has the disadvantage that the alignment process can only be verified by an interpretation process whereby an enterprise architect (or whoever has the job of making the alignment) determines what the reference architecture means, and argues for the alignment of their architecture to the model. This is a highly ambiguous and subjective task, and is prone to errors and even misuse.

A formal representation of a reference model addresses this problem by providing an unambiguous (or at least, less ambiguous) representation of the reference model, and allows for the definition of objective criteria for whether an architecture is actually conformant.

But a formal representation brings up a new issue; while some value can be gained from simple formal representations (such as the list of business areas, lines of business, and subfunctions of the BRM), most reference models have more complex structure than simple lists and hierarchies. Furthermore, description of how an enterprise architecture aligns with such a reference model architecture requires more complex consistency checking than is usual for a taxonomy.

Fortunately, 2004 saw the adoption by the W3C of the OWL standard for representing *ontologies*, which are formal models that allow the sort of complexity required by enterprise reference models. Furthermore, the OWL standard provides a formal semantic for the meaning of these models, which addresses the issues of fragility and ambiguity of informal models. Finally, OWL provides a framework for combining ontologies and checking their consistency, thereby providing the framework for a systematic discipline for determining how well architectures match the model.

[2] W3C stands for the World Wide Web Consortium

[3] Some of the FEA models are available in XML as well as in natural language. However, XML alone (not being a graph representation) can not describe all the relationships within and between the models. RDFS and OWL layered on top of XML provide us with all the language constructs needed to represent FEA.

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

2.2 Business Value of FEA RMO

The use cases outlined in section 4 of this white paper provide rich examples of how FEA RMO may be used. Simple use cases, such as some in the *Semantic Detective and Shared Concept* subject areas have already been implemented through model browsing and search capabilities shown in figures 3, 4 and 5. We are currently working on further enhancing them with special-purpose mapping and editing capabilities. *Legislative and Best Practice* advice can be delivered with the assistance of extra models (such as Balanced Scorecard ontology) linked to the FEA models. This option would include browsing, querying, as well as editing capabilities. We envision editing to be limited to adding information to the FEA models, such as agency extensions and mappings not modifying existing "gold" concepts.

The business reasons for pursuing more advanced use cases such as described for the *Component Search* and *Shared Concept* subject areas include the following:

- The use cases have been developed based on perceived needs of consumers of the FEA. They provide value that is actually required by the OMB in order to ensure alignment between agencies architectures, identify overlapping investments and ensure interoperability and re-use
- Agencies could start entering their enterprise architecture information or provide feeds from their existing enterprise architecture tools so that the information be merged and reasoned over
- A feature rich implementation of the *Legislative and Best Practices* use case area could provide information to agencies that is essential for them to be able to show compliance to legislative and regulatory requirements
- Architects in the agencies could comment on the models creating a community of practice[4] that can evolve and improve usefulness of the FEA.

The only drawback of implementing this plan is the commitment required to execute it. This commitment can, however, provide maximum pay off by satisfying critical needs currently unmet.

For example, today OMB takes three months to process exhibit 300 submissions. Once the submissions are processed, agencies are left with only one week to decide on collaboration opportunities suggested/identified by the OMB. Because the time frame is so short, this just does not happen and large potential savings are left unrealized.

Using FEA RMO to merge and reason over submissions would quickly identify most promising candidates for collaboration. Three months can be brought done to two weeks, leaving sufficient time for agencies to negotiate and implement joint initiatives, resolving overlaps and saving tax payers millions of dollars.

A staged implementation plan is also possible. For example, the work can begin with model browsing and mapping capabilities, aimed at enabling the use cases in the medium term, but in the short term aimed at gaining support for the FEA and the FEA RMO. With this support, more complete capabilities can be developed, as well as support for additional use cases. As the value of the models and the FEA in general is recognized, these efforts can be expanded.

Finally, we want to assert that it is highly unlikely that all the agencies will ever standardize on a single system. In fact, even within individual agencies there are multiple tools for describing and managing architectures, performing IT planning processes, measuring IT portfolios and other related tasks. This diversification is here to stay. Furthermore, an agency may want to leverage publicly available component registries and other technical repositories.

[4] Described in TopQuadrant SWANS proposal

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

Semantic Web standards are the only practical answer in this situation. They have been designed specifically to enable merging of distributed data with heterogeneous schemas. The work described here has proven that this promise is tangible and that business applications can be developed quickly.

2.3 FEA RMO Ontology Architecture

The FEA Reference Model Ontology architecture mirrors that of the FEA itself (see the FEA PMO web site for the FEA structure diagram); that is, the Performance Reference Model (PRM) organizes the overall architecture, making reference to the other models as needed, using the Bridge Axiom pattern described in section 5 of this paper. The Business Reference Model (BRM) in turn draws upon the Service Reference Model (SRM), Data Reference Model (DRM) and Technical Reference Model (TRM) as needed, using the same pattern. Each of these models is made in layers, which are expressed using repeated applications of the Class-Instance Mirror pattern (also described in section 5).

Performance Reference Model

The PRM is organized into layers called Measurement Areas, Measurement Categories and Generic Indicators. Each of these layers is implemented as a series (daisy chains) of Class-Instance Mirror patterns. This work used version 1.0 of the PRM, Volumes I and II.

Business Reference Model

The BRM is organized into Business Areas, Lines of Business and Subfunctions, implemented as a series of Class-Instance Mirror patterns. This work used version 2.0 of the BRM.

Service Component Reference Model

The SRM is organized into Service Domains, Service Types and Components, implemented as a series of Class-Instance Mirror patterns. This work used version 1.0 of the SRM.

Technology Reference Model

The TRM is organized into core Service Areas, Service Categories, Service Standards and Service Specifications, implemented as a series of Class-Instance Mirror patterns. This work used version 1.1 of the TRM.

Data Reference Model

The DRM is organized into Business Context, Subject Areas and Super Types. The work used version 1.0 of the DRM.

FEA Core Ontology

The FEA RMO includes a model that is not explicitly called out in the FEA , where concepts and properties that are common to all the reference models are defined. This provides a modularity to the ontology design that allows for simplified maintenance and integration of the models.

Bridge Models

The FEA RMO architecture also includes a number of “bridge” models that describe the mappings between two other models. By convention, these are named by concatenating short names of the two models with the figure ‘2’ between; for example, the BRM2PRM model has been filled out to reflect the relationships between the PRM and the BRM as specified in PRM v2 vol. 1.

Currently, the bridge models are created only when there is a subclass relationship between classes in two ontologies. For example, the BRM Subfunction is a subclass of the PRM Measurement Indicator or the BRM Subfunction is a subclass of the DRM Business Context. Mappings between ontologies implemented through Object Properties (e.g., *srm:Component runsOnPlatform trm:SupportPlatform*)

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

have not been separated into their own bridge models. Technical reasons for this approach are explained in more detail in section 6 of this paper (Tooling Issues).

Government Core

FEA reference documents contain a number of concepts that are "used by" the FEA models, but are not directly modeled or described by them. For example, one such concept is "Government Agency". Others include initiative, committee, and policy. We have created a model separate from but complimentary to FEA models to hold this information.

Agency Enterprise Architectures

Agencies are creating their own BRM, PRM, SRM, TRM and DRM mapping and aligning them with the FEA. FEA RMO architecture accommodates agency specific models. We have developed DoD SRM and DoD TRM as a proof of concept for this approach. Since DoDAF (DoD Architecture Framework) is a key part in describing DoD Enterprise Architectures, the work on DoD SRM has motivated us to develop DoDAF ontology. It now exists and is connected to the FEA RMO through a bridge model of its own.

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

3. Assessment of the Semantics of the FEA

One of the advantages of a formal model over a natural language model is the reduced likelihood of ambiguity. The formal nature of the model provides strict interpretations of what the relations between items is meant to be. In a natural language model, stylistic distinctions could be interpreted as having deep meaning ("sometimes it says that a committee is *part of* the senate, and sometimes it says that it *reports to* the senate. Which is it?"). In a formal model, relationships are spelled out explicitly; even the fact that there might be several words for the same thing is represented explicitly.

But this precision comes at a cost. In order to develop a formal model all the specifics need to be defined and the ambiguities resolved – nothing can be left to the fuzzy process of reader interpretation. In the process of changing an informal model into a formal one, the modeling team will have to work out any ambiguity or lack of clarity in the description. This can be quite educational for the modelers, and the results of their education become available in the form of the formal model, but still the product is only as good as this process.

Fortunately, in the case of the FEA models, the natural language descriptions were, for the most part, quite well-organized and consistent in terms of style and approach. Nevertheless, the process was not without its challenges. We used an alternatives approach to our design, outlining the different interpretations (and how they would be reflected in a formal model). We will not present the alternatives here, but highlight some of the issues that were raised by this analysis.

3.1 Inconsistencies

Logically speaking, "inconsistencies" refers to a logical contradiction in an axiom space. This condition can be a reflection of any of several different issues. Here, we use a looser notion of the word, to refer to times when different parts of a model seem not to conform to similar patterns.

For example, throughout the FEA RM, levels of hierarchies are named; for example, the TRM has the four tiered structure of *Areas*, *Categories*, *Standards* and *Specifications*. However, at some points there was need for deeper structure. Rather than adding a new level, or re-arranging the information to fit into the current levels, the architecture uses repetition within names to imply structure, as shown on p. 43 of the TRM v 1.1, with the replication of "Database Access" and "ORB".

3.2 Conflicts

We use the word "conflicts" to refer to situations in which the models seem to imply contradictory statements, that result in confusion when aligning an architecture to the reference model. An example of this can be found in the TRM.

In the TRM (p. 10), we are introduced to the term ***Technologies,*** which "refers to a specific implementation of a standard within the context of a given specification."

The TRM goes on to give examples of the term, including "PL/SQL is an Oracle implementation of the SQL Standard" and "ISQL/w is a Microsoft implementation of the SQL Standard."

The TRM continues (p.11) to define the term, "***Standard,"*** giving "Programming language standards" and "Database Management System standards" as examples.

The above would make us to conclude that PL/SQL is a Technology and SQL is a Standard and Oracle is a Vendor.

Yet on page 30, TRM says that "*Database*" is a Standard and "Oracle" is a Specification, and on page 44, it says that "Middleware" is a Standard and "*Database Access: PL/SQL"* is a Specification. These sorts of inconsistencies make it difficult to decide how to align a particular architecture with the reference model.

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

3.3 Omissions

Omissions in a reference model can be quite subtle; for example, a connection between models can be indicated, but the specifics of the connection are not given. An example of this comes from the interaction between the PRM and the BRM. Other omissions found within FEA have to do with introducing certain concepts and frameworks without making it clear how they relate (or fit in) to the main structure of the models. By the main structure we mean the hierarchy of levels described in section 5.4.5.

Omission – Match between PRM and BRM

On p. 16 of the PRM v 1.0 Volume I, we find, "As with Mission and Business Results, use of the Processes and Activities Measurement Area should use the BRM as the starting point. The BRM includes a Mode of Delivery Business Area that is designed to identify at a very high level the process that is being used to achieve an intended purpose. The Measurement Indicator(s) agencies choose should be an extension of the Mode of Delivery the IT initiative aligns with."

This may lead us to decide to make Model of Delivery Business Area a subclass of Process and Activity Measurement area, following the “bridge axiom” pattern below, just as we did for the Mission and Business Results Measurement Area. Correspondingly, subfunctions become indicators. But, unlike the case of the Business and Mission Results Measurement area, PRM lists categories and indicators for this Measurement area that do not correspond to areas, categories and subfunctions of the BRM Mode of Delivery. They are:

- Financial - Achieving financial measures, direct and indirect total and per unit costs of producing products and services, and costs saved or avoided;
- Productivity & Efficiency – The amount of work accomplished per relevant units of time and resources applied;
- Cycle Time & Timeliness - The time required to produce products or services;
- Quality - Error rates and complaints related to products or services;
- Security & Privacy - The extent to which security is improved and privacy addressed; and
- Management & Innovation - Management policies and procedures, compliance with applicable requirements, capabilities in risk mitigation, knowledge management, and continuous improvement.

These areas do not correspond to any level of the BRM; nor do the indicators correspond to subfunctions. There is some similarity, but the linkage between the two remains implicit.

Omission – TRM Standards Types

On p. 11 of the TRM document, twelve different categories of Standards are listed. These categories are not used elsewhere in the document; no Service Standard is associated with any of the categories.

Furthermore, items that are normally thought of as “Standards” (and which would fit intuitively into these categories), like SMTP, HTML, TCP/IP, etc., are listed in the model not as Service Standards but as Service Specifications. Is it simply an omission, that the standard category of all these things has been left out, or is it an inconsistency, that the standards appear as specifications? In any case, the categories on page 11 are not cross-referenced with anything else in the document.

We conjecture that the reason for this inconsistency/omission has to do with the failure of the strict functional hierarchy structure used for the TRM to cover the subtleties of the situation. There are several ways in which standards can be organized. Furthermore, depending on the technical area, they can have a different meaning and be realized at different levels of detail. A single hierarchy cannot easily capture

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

this structure. The authors of the TRM included the alternative way of looking at standards, but were unable to link them in a consistent way into the main hierarchy of the TRM.

Omission – SRM Component Types and connection to other models

Figure 4 on p. 7 of the SRM document introduces the "conceptual hierarchy of components":

- Federated Component
- Business Component System
- Business Component
- Distributed Component

It says:

"The SRM is decomposed into lower levels of granularity beginning from the process and application level to the software and component and module level. This level of decomposition provides various perspectives for stakeholders and solution architects to support the adoption of components and services within an IT initiative, asset or investment."

It is not clear, however, how this hierarchy relates to the main hierarchy of SRM with its 7 Service Domains, 29 Service Types and 168 Service Components. After describing the conceptual hierarchy on pp. 6 and 7, the document never returns to it.

Do Service Domains correspond to the highest level of granularity – Federated Components? This would mean that Service Types are intended to represent Business Components Systems and Service Components represent Business Components leaving Distributed Components not represented.

Or are these alternative hierarchies totally orthogonal to each other? This could simply mean that when a component provider describes a component, he should classify it according to one of 168 SRM Service Components as well as specify its level of granularity as defined by the Figure 4 of the SRM.

On p. 28 the SRM document gives an example of a connection between the SRM, the PRM and the TRM in the following way:

- Service Component *supports* Performance Measures from the PRM
- Service Component *accessed* using Access Channels defined in the TRM

One of the questions raised by the example: "What are other connections between the SRM and the TRM?" Access Channels is just one of the Service Categories described in the TRM. Do all of the TRM Service Categories connect to components or only some? For example, what is a relationship (if any) between a Service Component and the Software Engineering Category of the TRM?

Furthermore, the Access Channels listed in the example on p.28 of the SRM are Web Service and Portal. We have noted that "Portal" does not correspond to any Access Channels defined in the TRM. TRM Access Channels are Web Browser, Wireless/PDA, Collaboration/Communication (e-mail, fax and kiosk) and Other Electronic Channels (System to System, Web Service and URL). "Portal" is not defined in the TRM. Portal Server is defined. It belongs in the Delivery Servers Category of the TRM together with the Web Servers, Media and Application Servers. This is probably just an error rather than the omission. We have listed it here because the error made it harder for us to understand the relationship between models.

A related question deals with the restrictions and consistency between relationships. TRM, for example, has a Category "Supporting Platforms". Members of this category are Mobile/Wireless, Platform Dependent and Platform Independent. It also has a category "Business Logic". Members of this category are Platform Dependent and Platform Independent.

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

If it is known that a component is accessed using Wireless/PDA, does it mean that the supporting platform for this component must be Mobile/Wireless? If it is known that a component's supporting platform is Platform Independent, but its Business Logic is Platform Dependent, should it be triggered as an error?

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

4. Use Cases of the FEA RMO

A number of use case subject areas have been identified for the FEA RMO. The current OWL models for FEA can be used to build system support for the use cases. For select subject areas we have flashed out detailed use cases. We have also run queries against the model to validate that it can support these use cases. Finally, we have a system, called FEA Registry, which supports some of the use cases.

4.1 Use Case Subject Area: Line of Sight

Line of Sight is described in the PRM volume 2. The current PRM ontology includes Measurement Areas, Measurement Categories and Measurement Indicators, and the links between them, as outlined in FEA PRM Vol. 1. It also includes a model of the generic value chain from technology initiative, through process and activity, to Business and mission objects and Customer benefits, as outlined also in FEA PRM vol. 1. The linkage of the BRM lines of business as measurement categories and indicators for business and mission objectives is also achieved through OWL/RDFS connections between the FEA PRM ontology and the FEA BRM ontology.

Current Status

The Current PRM Ontology provides instances corresponding to the five measurement areas where the performance of a technology initiative can be measured. A particular initiative can be mapped to these instances according the questions given in PRM Vol. 2 figure 2. Following use cases have been identified:

- Identify Initiatives supporting certain performance measures in a connection with a specific Line of Business
- Identify how a specific component or technology support specific measurement indicators
- Report component usage across business subfunctions

4.2 Use Case Subject Area: Component Search

The use cases described in this subject area assume a heterogeneous environment where a number of registries and other information sources may co-exist. FEA RMO based system can merge the data producing answers based on the combined knowledge. Similarly to how crawling and creating search indexes for the web works, the discovery and merge of the data can be done on a periodic basis.

Some of the use cases in this subject area are similar to the use cases in the *Line of Sight* in terms of steps performed. Actors are different though. In this subject area they are mainly IT, in the Line of Sight they are predominately business people. The intents are somewhat different as well.

4.2.1 Name/Goal: Identify re-usable component

Actor: Enterprise Architect

Stakeholder: OMB, Agency that Enterprise Architect works for.

Precondition:

- Service component type known (for the purposes of use case validation, we have used as the test data Reservation/Registration component for grant search/registration called One Stop, One Form - this is an example from the OMB *FY06 FEA Additional Instructions* document)
- Business subfunction(s) supported by the component is known (for testing purposes we assumed "Elementary, Secondary, and Vocational education")

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

- Technical characteristics of the component (for example, access channels) are known (for testing purposes we assumed Web Browser)
- FEA RMO based system has access to the sources of information (for example, exhibit 300 information, component registries and repositories in the various agencies, enterprise architecture models in the various agencies)
- The sources of information are either OWL data based on the FEA RMO or other sources that have been "mapped" to FEA RMO

Steps:

1. Enterprise Architect composes the query
2. Enterprise Architect asks a system to run the query
3. A system accesses and merges data about components from the different sources
4. The system applies the rules identified by the OMB – components are likely to be re-usable if they implement the same service as described by the SRM, support the same Lines of Business/SubFunctions as described in the PRM, use the same or compatible technology as described by the TRM, has similar measurements as described by the PRM.
5. The system comes back with instances of components and the initiatives that implemented them
6. Enterprise Architect selects one of components and asks to see what business functions it supports
7. System returns required information
8. Enterprise Architect repeats steps 5 and 6 until he identifies a suitable component

Postcondition:

Enterprise Architect knows if there is a re-use candidate component

4.2.2 Name/Goal: Determine technology choices for a component

For the purposes of describing use case steps and for running validation queries we have used a real time collaboration component.

Actor: Enterprise Architect

Stakeholder: Agency that Enterprise Architect works for, IT Department of the Agency

Precondition:

- Service component type known and exists in the FEA
- FEA RMO based system has access to the enterprise architecture models of the various agencies
- Enterprise architecture models are either expressed in OWL based on the FEA RMO or are in other systems/representations that been "mapped" to FEA RMO.

Steps:

1. Enterprise Architect composes the query to find technology options for real time web collaboration (communication service area in the SRM) that have known performance characteristics in the area of reliability and user satisfaction (these are Generic Measurement Indicator Groupings for Technology Measurement area)
2. Enterprise Architect asks a system to run the query

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

3. A system accesses and merges data about technologies from the different sources For example, one agency has used Webex, another has used Raindance, another agency uses NetMeeting. The system knows they are used to support delivery of realtime collaboration (Real Time / Chat in the SRM). Even though one agency used mean time between failure (MTBF) to measure reliability and another used rate of fault occurrence (ROCOF), the system is able to recognize that they both measure Reliability (from the PRM).
4. The system comes back with the list of products that have been used to deliver specified capability, agencies that use them and performance indicators.

Postcondition:

Technology options identified or no matching options found

4.3 Use Case Subject Area: Shared Concept

The FEA Reference models can assist enterprise architects from different agencies to understand when they are "on the same page" in use of terminology. For example, TRM mentions "Video Conferencing" as a service standard, SRM talks about it as a component. A key step of being on the same page is knowing that there are distinctions and when one talks about "video conferencing", he could mean more than one thing. The FEA models provide a reference by which architects can disambiguate their terms by referring to the various usages of a term in the models; one could refer to the "service component" sense of video conferencing.

A more advanced use cases in this category can assist OMB in identifying collaboration opportunities between IT initiatives.

Current Status

The current FEA RMO provides detailed and systematic control over this sort of terminology variation, and makes clear the situation in which a single term is intentionally used in two different senses (e.g., the lines of business from the BRM also appear as measurement categories in the PRM).

FEA RMO can fully support reasoning and identification of collaboration opportunities.

4.3.1 Name/Goal: Identify collaboration opportunity

Actor: OMB Analyst

Stakeholder: OMB, United States taxpayers.

Precondition:

FEA RMO based system has access to the sources of budget information such as exhibit 300 information

Steps:

1. Enterprise Architect asks a system to run the query to identify IT initiatives that are most likely to "overlap" and have high collaboration potential
2. A system accesses and merges submission data from the different sources
3. The system applies the rules identified by the OMB – IT initiatives have high collaboration potential if they implement the same components as described by the SRM, support the same Lines of Business/SubFunctions as described in the PRM, use the same or compatible technology as described by the TRM, has similar measurements as described by the PRM.
4. The system comes back with the list of complimentary initiatives and potentially overlapping components

Postcondition:

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

OMB Analyst has a “top hit” list that he can now analyze in more detail

Exploratory sketch for this use case is shown in the screenshot below (from TopQuadrant’s *FEA Capability Manager* work in early 2003).

Figure 1: Using FEA RMO to identify partnering opportunities

4.4 Use Case Subject Area: Legislative and Best Practices Drivers

The value of any reference model is that it brings uniformity to projects that are performed by different groups. It also provides advice to designers of solutions as to the components of their solutions that they might otherwise overlook. The reference models of the FEA are no exception. In this use case, the reference models (in particular, the PRM) are used to provide best practices advice to agency program managers as they plan the performance evaluation of their IT solutions.

Program managers will want to make use of other methodologies (such as Six Sigma, Balance Scorecard, etc.) in their analysis of IT project performance. The measurements also are influenced by a variety of legislative acts such as Clinger-Cohen. The FEA models can help align these efforts, or at least make it possible for program managers to understand what aspects of their program correspond to activities in other programs. In order for the PRM to provide guidance to these projects, there must be a mapping of these methods to the PRM. For example, what is the relationship between the Balanced Scorecard and FEA?

Current Status

To fully support use cases in this subject area, we will need more comprehensive information on legislative and best practice drivers than currently exists in the FEA models. We have, however included the following constructs for a limited support of the relevant use cases:

- A class BusinessDriver with subclasses BestPractice and LegislativeRequirement
- Instances of the above subclasses identified in the PRM

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential. Unpublished Property of TopQuadrant

- A relationship (drives) between the individual best practices or legislative requirements and relevant measurement areas

The diagram that follows shows a partial view of an ontology graph that can support queries to identify a relevant business driver for a given measurement indicator. In this diagram:

- Black boxes indicate classes of things; red boxes indicate individual instances. The 'MA' prefix is used for instances of a class MeasurementArea.
- Red lines with an 'io' (instance of) relationship connect instances to their respective classes.
- Black lines with an 'isa' relationship show connections between subclasses and their parents.
- The many blue lines are relationships among instances.
- Beyond the right hand boundary of this graph are measurement areas for process and activities, customer results and business and mission results. For example, "*MA_Technology providesValue to MA_ProccessAndActivity*".

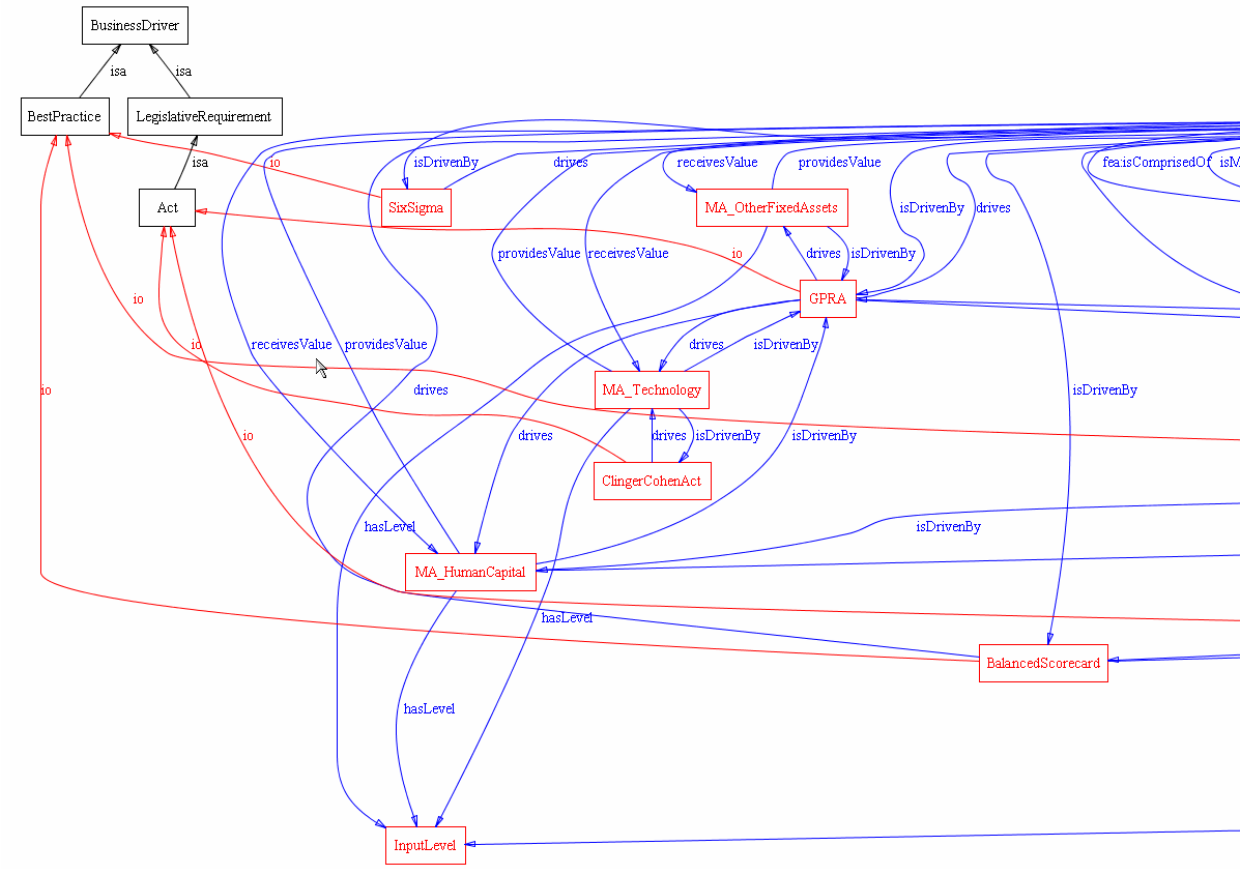


Figure 2: Ontology Graph for Best Practice and Legislative Drivers

In looking at this small piece of the model the following points become evident:

- The model has rich relationships that "connect the dots" between concepts stated in the FEA
- These relationships provide means through which the model can be understood and reasoned over
- Using this power, compliance to the FEA can be determined, merges can be performed and the line of sight capability realized

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

4.5 Use Case Subject Area: Learning and Understanding FEA (aka Semantic Detective)

The FEA reference models are written in natural language, authored by committee. While this provides the FEA models the benefit of multiple viewpoints and easy accessibility, this method of dissemination has the drawback that any use of the reference models is subject to the interpretation of the reader of the model. In the worst case, reference models that are disseminated only as natural language documents can be reduced to architecture Rorschach-blots; each reader interpreting them in any way they choose. The current FEA mitigates this effect with systematic and objective structures (e.g., Appendix A of PRM vol. 1).

The discipline of rendering a reference model in a formal modeling language (such as OWL, though similar benefits accrue from other modeling formalisms like UML) can uncover inconsistencies and incompleteness in the reference model. A uniform semantics for a formal model can also aid in communication between different modeling committee members, by making explicit parallels (and lacks of parallels) between different parts of the models.

Current Status

The basic support for this use case can be provided by modeling tools like Protégé. The formalization process itself provides checks on the coherence of the model. Very simple considerations that are troublesome to confirm in a natural language document can be explicitly stated in the model.

For instance, the five measurement categories listed on page 15 of the PRM vol. I should be the same as the headings in the "Category" column in the beginning of Appendix B. This is indeed the case, though this correspondence is implicit in the document. In the model, this relationship is explicit, and any gap on either side is readily visible.

Similarly, the table in the PRM vol. I, Appendix A mirrors the structure of the BRM. The measurement category "Defense and National Security" does not appear in the BRM itself, but in the DoD supplement to the BRM. In fact, the only place where all the BRM lines of business appear in a single table is in the PRM (where they are called measurement categories)!

4.5.1 Use Case Name: Browse FEA Models

Actor: Enterprise Architect

Stakeholder: FEA PMO, Agency that Enterprise Architect works for

Goal: Gain better understanding of the FEA

Precondition:
There exists a system that allows browsing and exploration of the FEA RMO

Steps:

1. Enterprise Architect asks a system to show different viewpoints into FEA models
2. A system returns viewpoints such as categories structures present in different models (for example BRM: business area – LOB – subfunction) or connection between models or alternative views into models (for example, component granularity for the SRM or input – output – outcome for the PRM)
3. Enterprise Architect selects a viewpoint (for example the "connection" viewpoint)
4. Systems presents viewpoint information (for example, PRM is connected to the BRM, BRM is connected to the DRM, SRM is connected to the TRM and their nature of connection - for example, component from TRM can have access channel from TRM)

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

Postcondition:
None

Outcome:
User viewed and inspected any supported perspectives on the FEA

As mentioned above, Protégé can offer some support for this use case. However, our experience have shown that displaying models as an FEA browser offers significantly more effective learning and exploration environment as shown by the screenshots below.

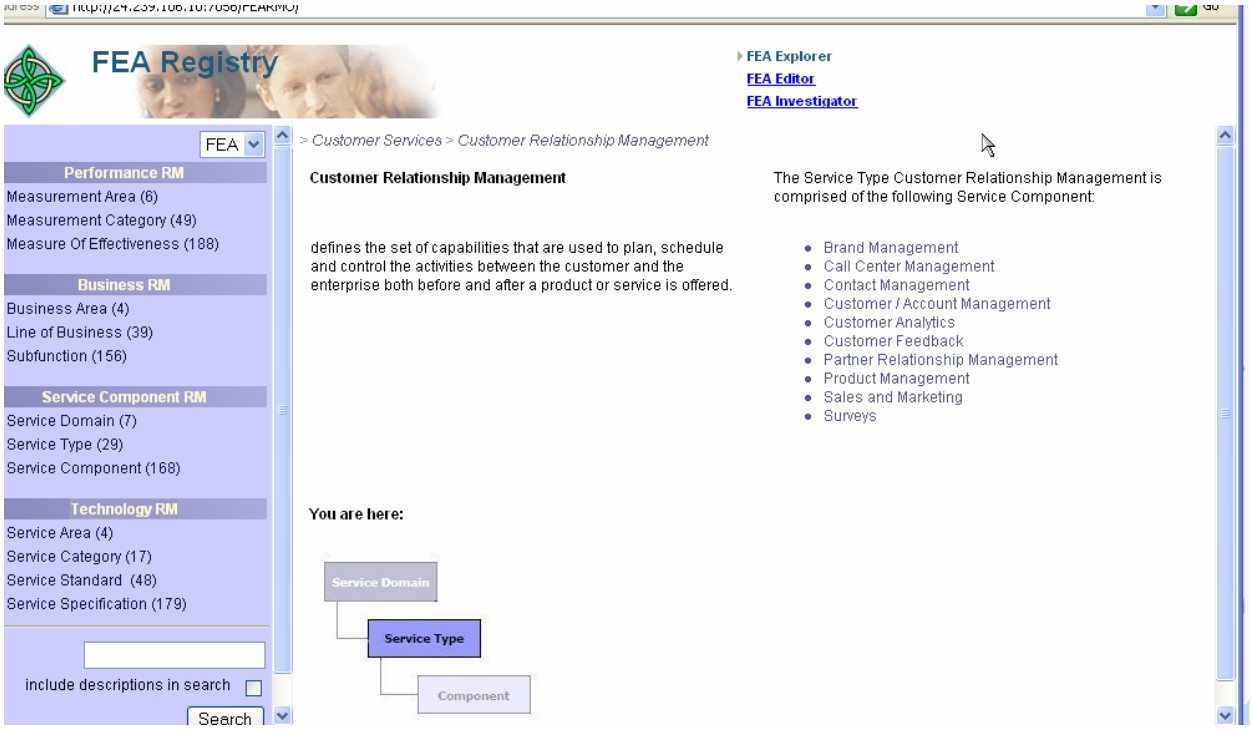


Figure 3: Exploring the SRM

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

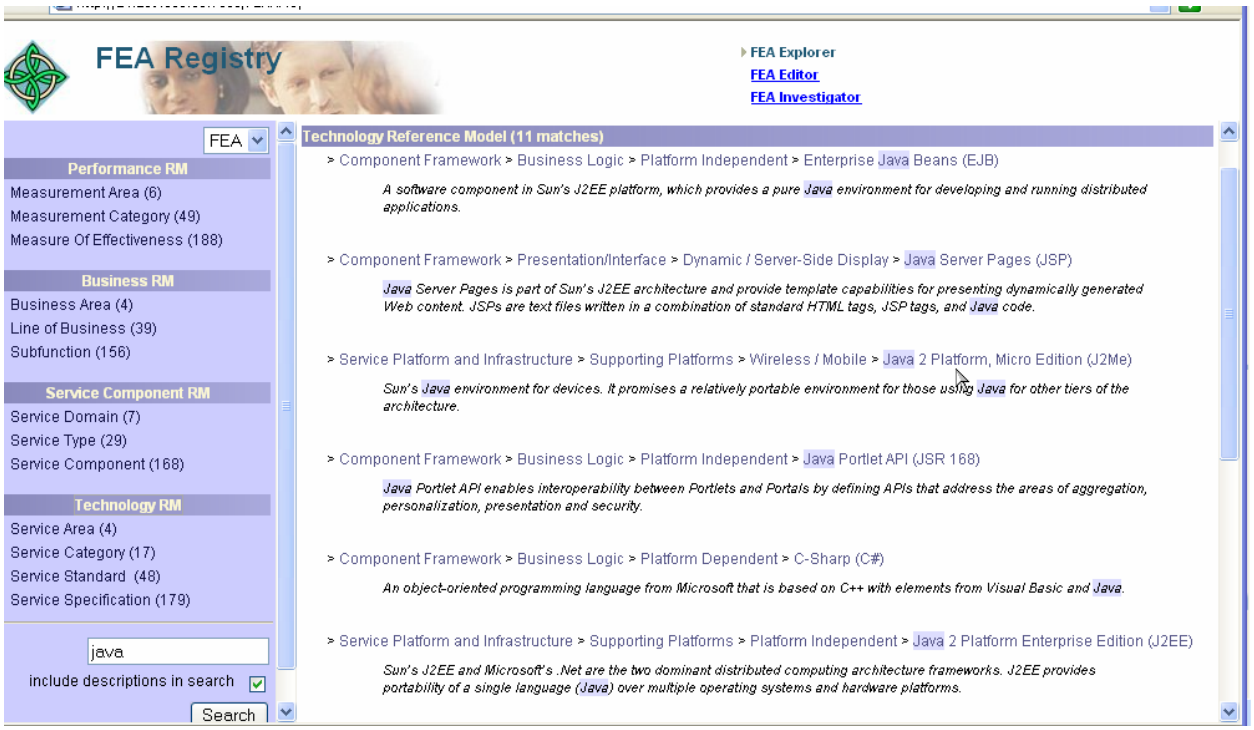


Figure 4: Searching the FEA

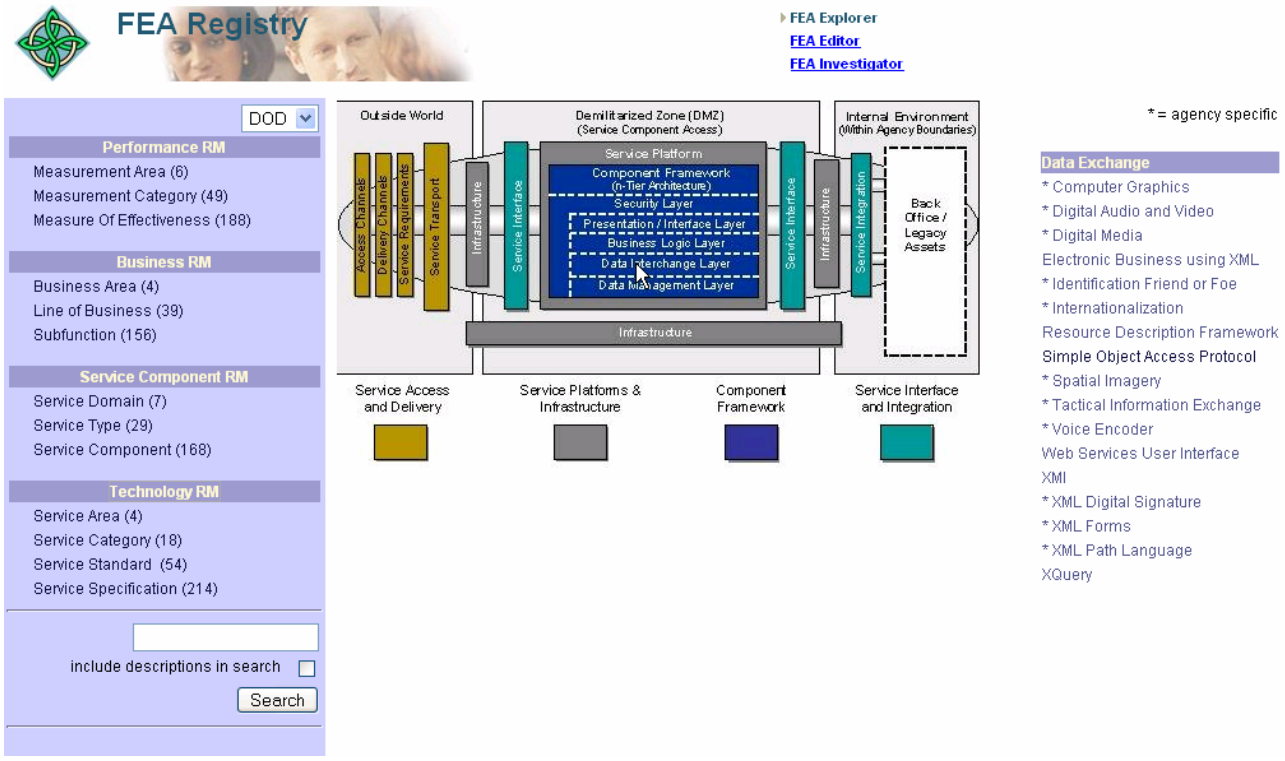


Figure 5: Exploring the DoD TRM

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

4.6 Use Case: Alignment Maturity Assistant

A reference model provides assistance and guidance to an enterprise architect through a process of "alignment" – that is, a particular enterprise architecture is "aligned" with the reference model by identifying corresponding parts of the architecture in the reference model. For an informal model, the process involves interpretation of both the reference model and the enterprise architecture, in a process that can be compared to legal rhetoric; the enterprise architect makes an argument that defends why they think that the architecture is compliant.

In a (formally) ideal world, the alignment process could be formally defined as follows:

- Each enterprise architecture is defined in OWL
- Alignment of the architecture with the reference model is specified using OWL connectives, like rdfs:subClassOf, owl:sameAs, owl:inverseOf, etc.

Alignments specified in this way bring considerable advantages; first, they are unambiguous. It is clear what maps to what, and in what way. They also provide for automatic validation of the mapping; if a contradictory mapping is made (for instance, the same component is used in two different ways that the reference model states are incompatible), then an OWL reasoner can signal the contradiction. Furthermore, an OWL reasoner can find connections between different mappings, finding synergies or redundancies between them.

This vision forms the basis for a maturity model for enterprise architecture. For example:

- Infant – ad-hoc architecture, without a documented plan.
- Child – documented architecture with informal FEA RM alignment
- Maiden – formally defined architecture, without formal mapping to FEA RMO
- Matron – formal architecture, mapped to FEA RMO, with some inference support
- Sage – Formal architecture, OWL-validated with the FEA RMO.

With such a maturity model in place, the goals of the Federal Enterprise Architecture could actually be achieved. Agency architectures would genuinely be aligned with the vision of the FEA. Different agencies would be able to understand the lines of business and components of other agencies, without extensive re-tooling and re-training. Agencies would not be able to claim compliance without showing genuine mapping of their enterprise architecture to the reference model.

In turn, the agencies would gain the benefits of compliance; the federal architecture provides checks and balances to ensure that agencies are compliant with legislative and regulatory requirements. Interchange of information between agencies will be simplified, improving the capabilities of each agency to carry out their tasks.

TopQuadrant is currently working on a number of experiments and prototype solutions where FEA RMO can provide value. Some of these are demonstrated by the various screenshots in this paper. One additional example is worth mentioning in the relationship to the Alignment Maturity Assistant.

Using technology from the Magpie project of the UK Open University, we are able to analyze documents for their alignment with the FEA, using a color coded approach to highlighting specific concepts that are part of the FEA. This could be useful as a tool for agencies needing to analyze their existing enterprise architecture documentation for mappings with the FEA. It could also be used by the OMB to speed up assessments of alignments.

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

5. FEA RMO Development Approach

5.1 FEA RMO Guiding Principles

The first step in building an ontology is to determine what questions the ontology will answer. These are called "competency questions." In our case, we wanted the FEA RMO to reflect as accurately as possible the concepts and relations in the FEA reference models. These models have been developed primarily to assist in the task of "alignment" – determining how particular agency architectures relate to a common core.

The authors of the FEA models did a good job of making the models as simple and elegant as possible. Each model has a similar layered structure. The interpretation of the layers, while not the same throughout, is managed in a fairly uniform way. Our goal in the FEA RMO was to reflect this elegance. That is, we modeled the structured portions of the FEA RM, and as far as possible, avoided modeling things that while mentioned in the FEA documents, were not an essential part of this core, elegant model.

In particular, this means that models of government agencies, business processes, legislation, and other government structure are not modeled in FEA RMO. These are important concepts, but they are not part of FEA, and hence are not included in FEA RMO (except as needed to complete linkages described in the models).

Even with this policy, we still found that the modeling process involved considerable interpretation of the models, determining how they are intended to be used:

- Will some concept reflect a set of objects (hence it should be an owl:Class)?
- Or will it be reasoned about as an individual in its own right, in which case it should be an instance?

It is not out of the question for a single entity to play both roles in a single model; but OWL-DL[5] restricts the circumstances in which a single entity may be both a class and an instance.

Despite these issues, we anticipate wanting to use the standard reasoning functions of OWL to draw conclusions about the connections between the various FEA models. Since OWL-Full reasoning is, in principle, intractable, we have opted to restrict the models we create to OWL-DL.

In order to satisfy these goals, we used a number of design patterns to manage the modeling process.

5.2 Ontology Design Patterns

The FEA RMO formalizes various details of the FEA models. Fortunately, the FEA models are, for the most part, fairly orderly and uniform. This means that the same issues arise again and again in different situations. We have summarized solutions for some of the more common issues in the form of design patterns. Using the patterns insured the uniformity of the modeling approach. We realize that these patterns are not expressed in a formal pattern language. This work still remains to be done.

5.2.1 Pattern: Class-Instance Mirror

A recurring pattern in many of the FEA models is the need to refer to some entity sometimes as an instance, and at other times as a class. For example, in the PRM, all 6 Measurement Areas are related to one another according to the diagram on PRM, vol. 1, p. iii. The diagram shows how the value of an IT initiative is tracked by the various Measurement Areas. In this figure, the areas are treated as instances. On the other hand, throughout the PRM (e.g., p. 13), measurement areas comprise measurement

[5] OWL Language consists of three species: OWL Lite, OWL-DL and OWL Full. OWL Lite is a subset of OWL-DL. OWL-DL is a subset of OWL Full that guarantees tractability of inferencing.

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

categories, which include generic indicators, which are groupings of indicators. This indicates that areas, categories and indicators should also be treated as sets of instances or classes.

One solution to this problem is to simply view these entities as both classes and instances. However, this solution is not available in OWL-DL, where there are restrictions on such usage of instances and classes.

The design pattern we use for this situation falls squarely within DL, and is motivated by the questions (queries) we want to be able to answer with the model. This pattern is recommended by the W3C Semantic Web Best Practices Group (http://www.w3.org/TR/swbp-specified-values/) by the name, *"Values as individuals whose enumeration is equivalent to the quality"*. We will show the pattern by example.

In Figure 6, we see (among others) the Customer Results measurement area. It is related to other measurement areas as shown in the figure. However, it also includes several Measurement Categories. We need to be able to find all the measurement categories of a given measurement area (and vice versa). We also need to share properties among measurement areas.

FEA RMO representation (shown below) is as follows:

- There is an instance (of class MeasurementArea) called CustomerResultsMeasurementArea.
- There is also a class (subclass of Measurement Category) called CustomerResultsMeasurementCategory. The five Customer Results Measurement Categories are instances of this class (as would be expected from its name).
- To link the Customer Results Measurement Categories and the Customer Results Measurement Area, the pattern includes a property (with a domain CustomerResultsMeasurementCategory and a range MeasurementArea) called isMeasurementCategoryOf (and its inverse, hasMeasurementCategory). This object property directly relates the instances of MeasurementCategory to the instances of MeasurementArea.
- The final piece of the pattern ensures that just the instances of CustomerResultsMeasurementCategory are the ones that are linked to CustomerResultsMeasurementArea. This is done by placing an owl:hasValue restriction on the property isMeasurementCategoryFor at CustomerResultsMeasurementCategory.

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

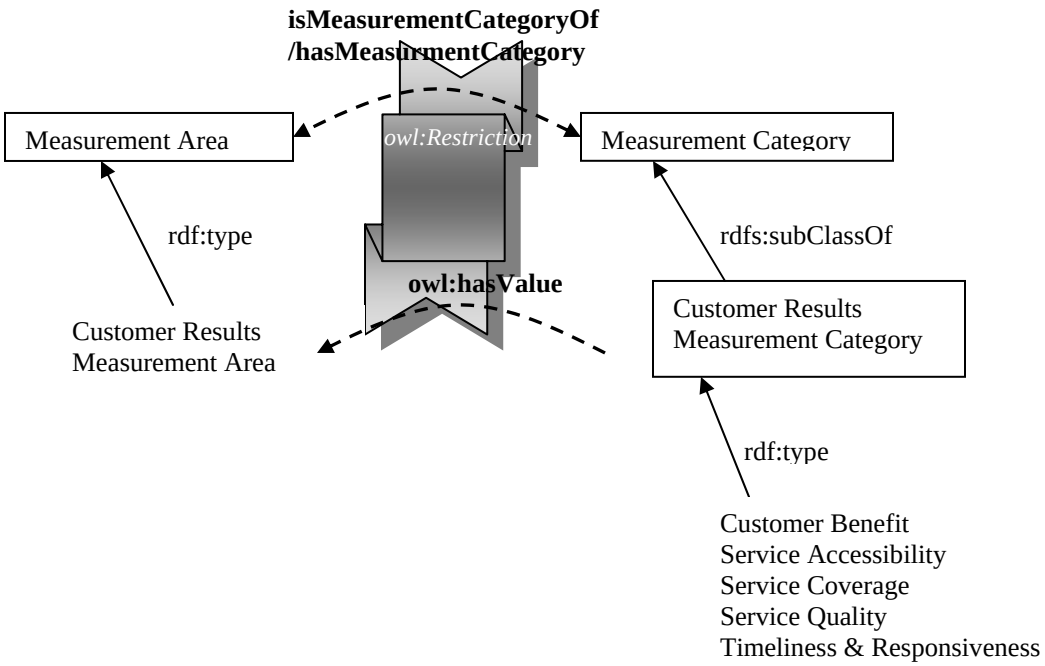


Figure 6: Example of the *Classification Enforcer* pattern

This pattern has the result that any Measurement Category that is an instance of the class Customer Results Measurement Category automatically *hasMeasurementCategory* Customer Results Measurement Area, and vice versa, thereby guaranteeing the consistency of the model. A useful side-effect of this is that all customer benefits, regardless of how they were specified, are available using a simple query for instances of CustomerResultsMeasurementCategory.

5.2.2 Patterns: Axiom Bridge and Axiom Bridge Model

It is common in the FEA models for one model to reference another; in these cases, an entity that played one role in the source model plays another role in the destination model. A good example of this appears in the PRM, where it references the BRM:

> *… the PRM's Measurement Categories are the same as the BRM's Business Areas and Lines of Business. … The Mission and Business Results Measurement Area is comprised of the following Measurement Categories:*
>
> - *The Lines of Business in Services for Citizens;*
> - *The Lines of Business in Support Delivery of Services; and*
> - *The Lines of Business in Management of Government Resources.*
>
> *-- PRM Version 1.0, vol. 1, p. 13*

The measurement categories above correspond to three of the four Business Areas in the BRM. Each is represented by a class in the FEA RMO BRM, whose instances correspond to the line of business for each area. The requirement of the quotation above is that the lines of business should be instances of the PRM class Measurement Category.

This is achieved by bridging the two models with a single axiom using rdfs:subClassOf. For example:

brm:ManagementOfGovernment rdfs:subClassOf MissionAndBusinessResultMeasurementCategory

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

Note that *MissionAndBusinessResultMeasurementCategory* is rdfs:subClassOf prm:MeasurementCategory.

As a result, all the instances of brm:ManagementOfGovernment (which correspond to the lines of business) are, according to the inference rules of RDFS, also instances of MeasurementCategory. This is depicted in the figure below.

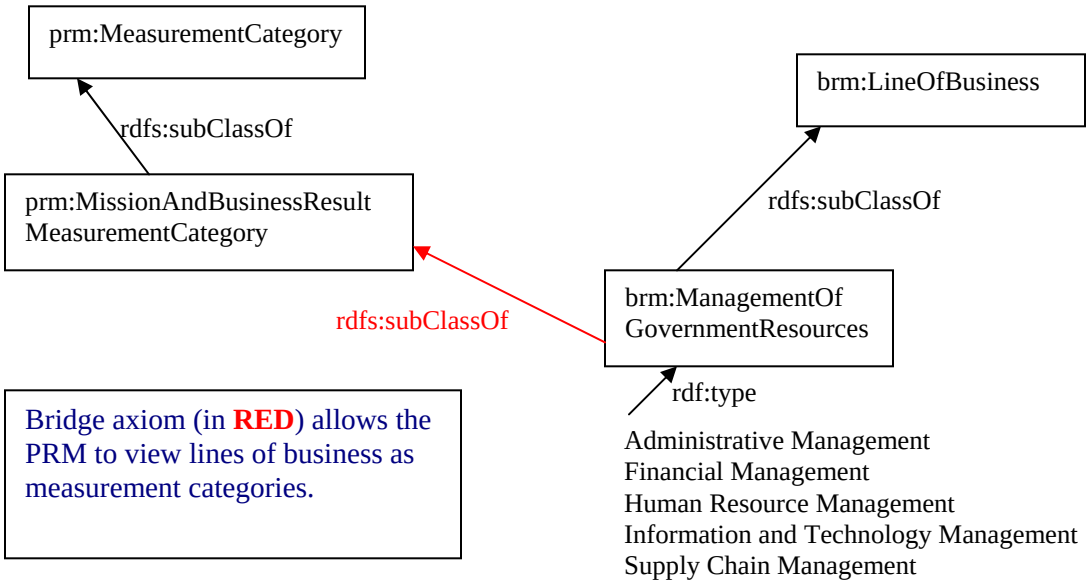


Figure 7: Example of the *Axiom Bridge* pattern

Using the Axiom Bridge pattern has the advantage (over the brute-force method of copying instances with the same names from the BRM to the PRM) that any updates to the BRM will be automatically (via RDFS reasoning) reflected in the PRM. It makes explicit the policy described in the quoted passage.

This pattern can be made even stricter by insisting on a *bridge model*, to hold the bridge axioms. This is what we refer to as Axiom Bridge Model pattern. The diagram for this pattern would look the same as the one shown in figure 2 with one exception. The red components are stored in a separate model file; they appear neither with the domain model nor with the range model. The bridge model then imports both the source models.

This approach has a number of advantages:

- It improves modularity, by allowing each of the source models to be self-contained; they needn't reference the other model directly at all. This means that different models could be "swapped in" as needed (e.g., later versions of the BRM).
- The modularity also makes it possible for the source models to validate as DL, independent of their larger context.
- Finally, this pattern works better with certain tools (see section 6 for a discussion of how Protégé works with model imports).

By combining patterns, we get some interesting advantages beyond what can be done by either of them alone.

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

In our example, if we apply the Class-Instance Mirror pattern to prm:MissionAndBusinessResultMeasurementCategory, then the combination of the Axiom Bridge and Class-Instance Mirror will automatically connects all of the Line of Business instances in the BRM to the appropriate Measurement Area instance in the PRM, without making any change in the BRM itself.

5.2.3 Pattern: Transitive Parent

A simple, but very common, pattern is to make a property that is not transitive a subproperty of a transitive one. A well-known example is to make a property called "hasParent" as a subproperty of "hasAncestor", where the latter is a transitive property.

The inferencial interpretation of rdfs:subPropertyOf is that any statement involving the subproperty also holds for the superproperty, though not vice-versa. That means that if [Harry hasParent Charles] then we can conclude that [Harry has Ancestor Charles]. And if [Charles hasParent Elizabeth] then [Charles has Ancestor Elizabeth]. This is the interpretation of rdfs:subPropertyOf defined in the RDF standard.

When we combine that with the definition of owl:TransitiveProperty, we get a very useful pattern. If P is an owl:TransitiveProperty, then we may conclude for any [a P b] and [b P c] that [a P c]. In the example above, if has Ancestor is a transitive property; we can conclude that [Harry hasAncestor Elizabeth] - as illustrated in the next figure:

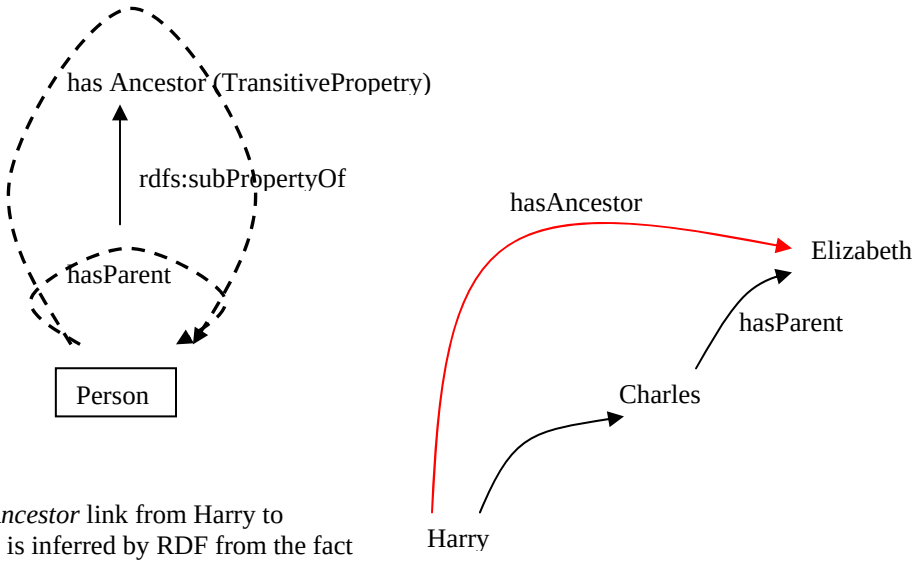


Figure 8: Example of the *Transitive Parent* pattern

This pattern is used frequently in the FEA RMO. Each reference model (PRM, BRM, SRM and TRM) defines specific relationships between its levels. For example, the TRM defines a relationship between ServiceCategory and ServiceStandard, and another between ServiceStandard and ServiceSpecification. But what is the relationship between ServiceCategory and ServiceSpecification?

By making each of these properties an rdfs:subPropertyOf a single property, which we call "isComprisedOf" (with an inverse comprises), and by making comprises transitive, and OWL reasoner can infer that a ServiceSpecification comprises a ServiceCategory.

By combining multiple instantiations of the Transitive Parent pattern, we can use a single transitive property to amalgamate information from several intransitive properties, chaining them all together.

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

This provides a number of important advantages. For example, since hasServiceCategory and hasServiceStandard are both subPropertyOf comprises, the code that displays all service categories for a service area can also be used for a page that displays all service standards for a service category. In both cases, the query will look the same because it uses the same object property – "isComprisedOf". This minimizes the code base and simplifies development, offloading the work to the reasoning engine.

However, if we want to create a new instance of ServiceStandard (let's call this instance x), we can not simply say that it comprises y, an instance of ServiceCategory. Because using *comprises* will not imply that x has an object property *hasServiceCategory* and hence, we can not infer that *x rdf: type Y_ServiceStandard*. This limitation would require us to maintain multiple class-specific insert queries.

But, with a small change to how we implement the Class-Instance Mirror pattern, a combination of the Transitive Parent and Class-Instance Mirror patterns can overcome this limitation:

- Instead of making Necessary and Sufficient condition on the owl:hasValue restriction to hasServiceCategory, we make the N&S conditions on the restriction to comprises.
- We also make a Necessary condition on hasServiceCategory restriction.

As the result we will be able to make the following inferences:

1. If it is asserted that *x hasServiceCategory y*, we conclude that *x comprises y*. Then the N&S condition is triggered and *x rdf:type Y_ServiceStandard* will be asserted.
2. If, on the other hand, we assert *x rdf:type Y_ServiceStandard*, then the Necessary condition triggers, and infers *x hasServiceCategory y*, as desired.
3. In addition, if we assert *x comprises y*, then the N&S condition is triggered, and we conclude that *x rdf:type Y_ServiceStandard*. Then, the Necessary condition is triggered, and we get *x hasServiceCategory y*.

The first two sets of inferences were possible before the change. The additional inferences described in the point # 3 are afforded by the change in the pattern. This inferencing is very convenient when creating instances under program control by allowing us to keep the program code generic - we don't need to know the specific name of the subProperty we are using.

5.2.4 Pattern: Common Data Property Enforcer

It is quite common for certain classes of objects to have some required pre-set indicator in a database or indexing system. For instance, all toll-free phone numbers in the US begin with 800 or 888; radio stations east of the Mississippi begin with "W"; in the Library of Congress, scientific works begin with "Q" (for the "category"). How can we specify that all the instances in a class must have some property in common?

One consideration for implementing this in a database is the way to enforce the required (restricted) value for each known instance. Another consideration can be a desire not to replicate this value (for example "Q") in the data store once for each known instance. In databases the implementation can amount to creating a "trigger" routine that returns the value whenever a query requires it.

One way to implement this with ontologies is through something called "meta classes". This involves creation of a Library of Congress Category class defined as a subclass of owl:class. We then give it a data property of "hasCategory". Scientific Works is then defined as an instance of this class with the value of "hasCategory" data property set to "Q". As an instance of a meta class, Scientific Works is also a class. Te instances of it are the actual scientific works.

OWL inference engines, however, would not automatically infer that each instance of Scientific Works gets its hasCategory data property set to "Q". We would need a database trigger equivalent to enforce

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

this. This could be done through custom code or by using Semantic Web Rules Language (SWRL), which is still under development, and a future engine that implements it.

Alternatively, we can use the native power of OWL by applying hasValue restriction to the hasCategory data property of the ScientificWork class. This precludes the need for a special-purpose trigger. The OWL keyword owl:hasValue indicates that any instance in the class has the specified value for a particular property. We also do not need the Library of Congress Category meta class.

The semantics of OWL specify that when this pattern is used, any query asking for the value of the property LOC:hasCategory for an instance of the Scientific Work class will get the value "Q". This is true even though the actual value "Q" appears only once; that is, in the triples shown in the pattern.

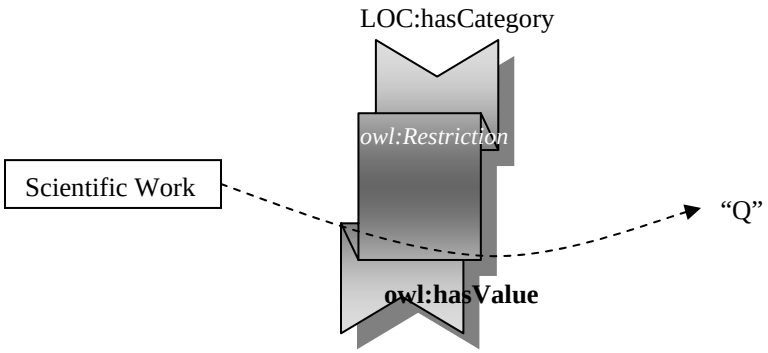


Figure 9: Example of the Common Data Property Enforcer pattern

Currently, FEA RMO uses this pattern only to enforce the value of "mnemonics" data property. For example, it can be used to ensure that a mnemonic for each instance of a Service Specification is "SSP".

We can not, however, use this pattern to enforce mnemonics for all the classes in FEA RMO. When a class has multiple inheritance in Object Oriented (OO) terms, multiple property values are enforced. This is true, for example, for the PRM Measurement Areas and the BRM Lines of Business:

- If the Data Property Enforcer is used to ensure that every BRM Line of Business has 'LOB' as its mnemonic, and
- The same pattern is used to ensure that every PRM Measurement Area has 'MA' as its mnemonic, and
- The BRM LineOfBusiness class is a subclass of the PRM MeasurementArea class then Measurement Areas would "inherit" two menemonics – 'MA' and 'LOB'

We put "inherit" in quotes here because the inferences that enable the pattern work backwards from what someone with OO background would expect. In OWL (and in other logic programming platforms) inferencing works up – from a subclass to its parent, not down.

This feature of the pattern may be quite useful in some situations, but in case of mnemonics we wanted to have a single mnemonic for a class. When a data property being enforced is functional, multiple inheritance described above, would raise an error. Therefore, combining Axiom Bridge and Data Property Enforcer patterns requires particular care and consideration.

5.2.5 Pattern: Daisy-chain

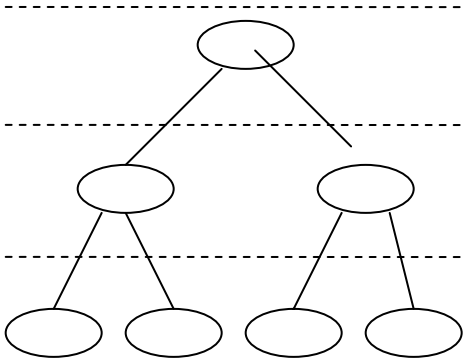
As we have shown previously, it can be useful to combine patterns. It is also possible to use one pattern in conjunction with itself, opening up the opportunity for unlimited combination of patterns. FEA RMO has such repeated use of the Class-Instance Mirror pattern. We call this a Daisy-chain pattern.

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

Consider the situation in which a hierarchy has a fixed number of named layers. This is common in taxonomy standards and thesaurus standards and is also true for all the FEA models. As shown in the table below the BRM, PRM, and SRM have 3 levels while the TRM has a fourth level, called Service Specification (not shown).

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

TRM	SRM	PRM	BRM	
Service Area	Service Domain	Measurement Area	Business Area	*Level 1*
Service Category	Service Type	Measurement Category	Line of Business	*Level 2*
Service Standard (+Spec)	Component	Measurement Indicator	Sub Function	*Level 3*



In each of the models, there is an "inclusion" relationship between elements at adjacent levels. For example, Line of Business belongs to a Business Area. However, there is also a need to talk about each entity as an element in its own right. For example, not only is each business area (the highest level) in the BRM treated as both, instance and class, but so do the lines of business (the next level down) because the PRM talks about all of the lines of business in a certain area. This means that we need all the entities in all models (at all levels) to be available both as instances and as classes.

Starting at the top, we apply the Class-Instance Mirror pattern repeatedly. For example four Business Areas in the BRM appear as both classes and instances, as shown in the figure below:

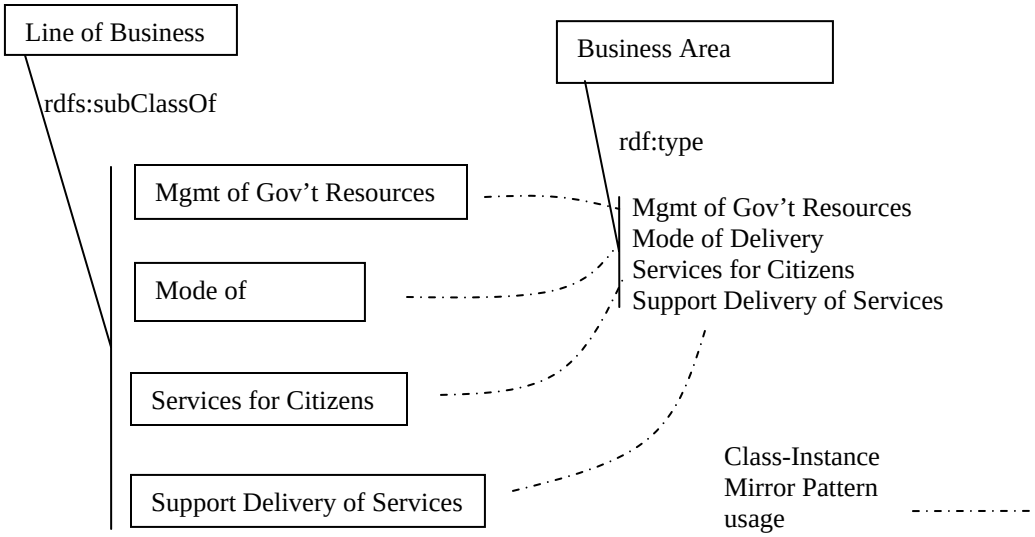


Figure 10: Example of the Daisy-chain pattern (1)

Since the Lines of Business also need to be treated as instances and classes, we repeat the pattern one more time between Lines of Business and Subfunctions, as shown below:

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

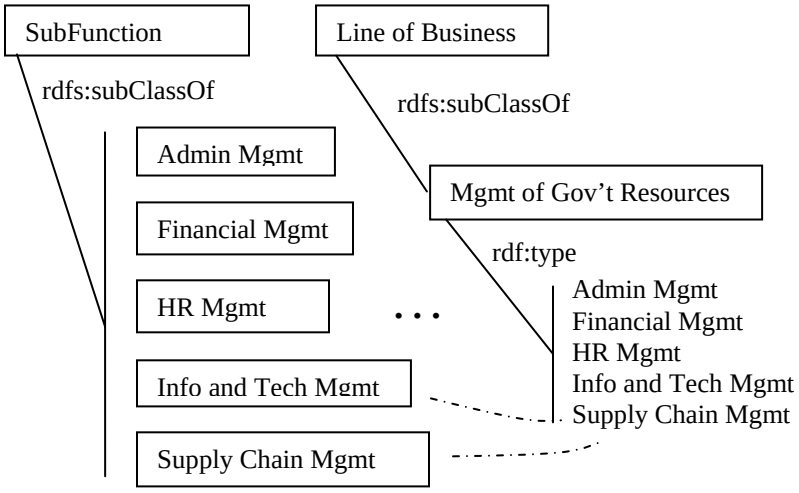


Figure 11: Example of the Daisy-chain pattern (2)

In the actual BRM, there are 39 lines of business in all, so there are 39 instances of the class Line of Business. Given the Daisy-chain pattern, this means there are 39 subclasses of SubFunction. While not called for in the Class-Instance Mirror pattern, we found it convenient to insert another level between the class SubFunction and these 39 classes to organize them, according to the business areas.

The following figure shows the final, completed model in Protégé. Classes flagged by Protégé in orange have defining conditions, and are the classes that participate in the Class-Instance Mirror pattern. Classes in yellow are either the base classes, or extra organizing classes. The last three business areas of SubFunction have not been expanded, in the interest of brevity:

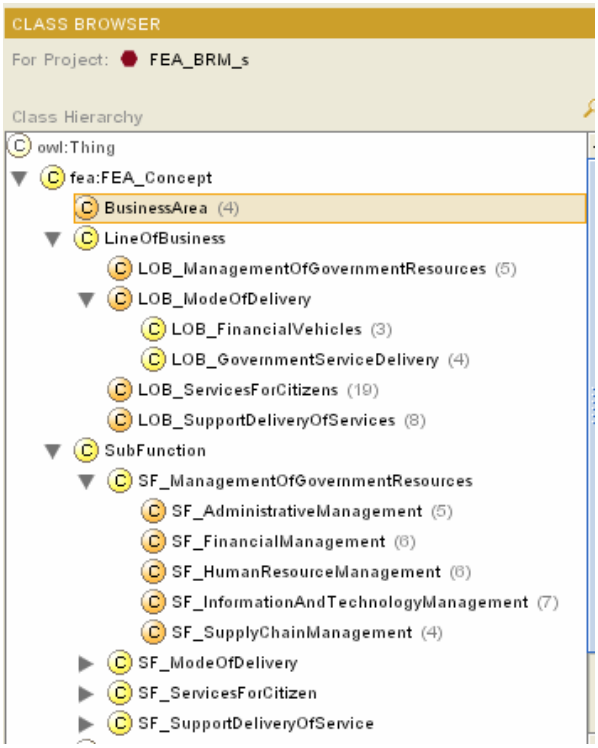


Figure 12: BRM Ontology in Protégé 2000

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

Let's revisit the competency questions that motivated this modeling style. In section 2.4.2, we see the following quote (for example) from the PRM:

> *The Mission and Business Results Measurement Area is comprised of [...] the Lines of Business in Services for Citizens*

Let's see how the daisy chain of Class-Instance Mirror pattern satisfies this (and similar) competency requirements.

Under the class LineOfBusiness is a subclass called LOB_ServicesForCitizens. Its instances are instances of LineOfBusiness. Specifically, they are the lines of business that comprise the Services for Citizens Business Area. This is assured by the hasValue restriction in the model. This means that the class LOB_ServicesForCitizens is exactly the class we need to connect to the Mission and Business Results Measurement Area to satisfy the requirement in this quote.

Four FEA models have been realized using this pattern-of-patterns, applied twice in the case of PRM, SRM and BRM (for three levels), and three times in the TRM (for four levels). This means that any query of the form "give me all the elements that comprise another element" can be satisfied by the (single) appropriate class in the model.

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

6. Tooling Issues

For the most part, the FEA RMO models have been developed in Protégé. Protégé is the most advanced modeling tool for the semantic web to date. Nevertheless, Protégé has its share of problems. We anticipate that many of the earliest users of these models will want to use Protégé, so even if we are willing to work with valid OWL files that happen not to be viewable with Protégé, we need to have a policy about how Protégé users should proceed.

For the remainder of this section, we outline some of the issues with how Protégé handles interrelated models that have a bearing on the use of FEA RMO with Protégé. We also identify the recommended way of working with FEA RMO models in Protégé.

We have increasing evidence that the Protégé team is serious about responding to the needs of the users of the Semantic Web standards. One example is the following posting (dated February 21, 2005) on the protégé-owl mailing list by Holger Knublauch, the main developer of OWL plugin for Protégé:

> *"I firmly believe that in order to make the Semantic Web real, it needs to exploit its Web nature. This means, it should be easy and natural to split large ontologies into smaller chunks and distribute these chunks in arbitrary places on the Web so that other ontologies can import and extend them.*
>
> *Protege currently only has very limited support for editing ontologies with imports: Only the top-level ontology can be edited, all others are grayed out and not editable. This is going to change though. We are as we speak working on support for editing multiple imports at the same time.*
>
> *Then it will be possible to edit a graph of ontologies, to make changes to either one, and to save the single files back to disc. This is work funded by the Darpa DAML program and we are trying our best to have a functional version of this available in April. Hard to promise anything because it's a non-trivial task.*
>
> *This should solve most of the issues, and no longer require the rather ugly work-arounds such as merging. I am not particularly motivated to work on temporary fixes, knowing that this whole mechanism will thoroughly change soon anyway."*

Until these issues are resolved, we have established an ontology architecture and the working processes described in this white paper that enables using Protégé to develop and maintain FEA RMO.

6.1 Import as Extension

Protégé allows (using the owl:import key) one model to refer to another. The anticipated scenario is that one model will import "base classes", and add new classes as subclasses of those base classes.

Consider, however, the following example using two ontologies:

- Dog Ontology that describes breeds of dogs
- Animal Ontology that has a class Canine and classes that describe Wolves, Jackals and Foxes

Protégé allows us to import Dog ontology into the Animal ontology, but Protégé UI will not let us make a Dog a subclass of Canine. Protégé disallows it even though, this sort of graph import is allowed in RDFS, and in fact, this is the way the relationship between the PRM and the BRM are structured.

Protégé allows us to import the Animal ontology into the Dog ontology and then make a Dog a subclass of Canine.

On a surface the second option combined with the owl:import statement in the Animal ontology, seems to accomplish what is needed. If we now open the Animal ontology, we will see that Poodle, for example, is

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

a subclass of Canine. This happens because the Animal ontology has owl:import statement for the Dog ontology. Protégé, however, does not like this situation and on save will corrupt OWL file for the Animal ontology by adding arbitrary statements from the Dog ontology in to it. We call this a "triple drift".

How does this relate to FEA models? In the quote given in section 5.4, the PRM states that the categories in the Mission and Business Objectives area are taken from certain classes of Lines of Business in the BRM. This follows the "Animal" pattern above - the PRM references the BRM, and places the BRM classes as subclasses of its own. In Protégé, the only way to do this is to:

- have the BRM import the PRM, and express the Lines of Business as extensions of the appropriate PRM classes
- don't have any references to the BRM in the PRM

Not only does this solution go against the organization of the FEA documents, it also violates the modularity of the models. The BRM is a resource that is re-used throughout the PRM, to organize various technical, performance and service entities. The "import-and-extend" pattern supported by Protégé insists that only the BRM can import PRM concepts, and specify the Lines of Business as extensions of all the appropriate classes. This means that the user looking at the PRM will not be able to see the measurement areas defined by the BRM Lines of Business.

One could ask, why not just change Protégé to fix this problem? We have reported the problem to Protégé developers and are hopeful that the fix is forthcoming. However, this issue may not necessarily be a bug in Protégé, but a reflection of a modeling strategy. Protégé is a modeling tool with its own rules, not a semantic web integration tool. Nor does it support all valid OWL files, only files that comply with its rules. As a modeling tool it currently takes a specific point of view (shared by OO models) that a reusable entity must live high-up in a class hierarchy. The only sensible way to import is to import high-level classes, and extend them, eventually creating instances of the extension classes, which will inherit properties of high-level classes. In this view of the world, importing subclasses (and doing any circular, cross-referencing imports) is nonsensical.

This world view is different from the one evident in the Semantic Web standards (RDF/S and OWL). It is also different from a view point common in Library Sciences as represented in thesauri work and standards.

There are some bugs still outstanding in Protégé (as of OWL plugin rel. 225) in which behavior under these circumstances is incorrect. Given the original design goals of Protégé, these bugs are not very surprising, since circular imports border on violating the usage parameters of Protégé.

6.2 Graph import

Since Protégé produces and reads files in OWL format, it is possible to get around the limitations on imports placed by the Protégé environment by including appropriate triples right in the OWL files themselves. In the "Animal" example above, the modeler can put in the triple that says that Dog is a subclass of Canine right into the Animal model, by using the OWL XML syntax. This is fine from a semantic web point of view, and Protégé can read and display the resulting OWL model with no difficulty.

The problem comes when Protégé tries to write such a model. Because of the design assumptions of Protégé, it brings select information about the subclass into the importing model. This results, from the point of view of the modeler who added the triples "by hand" into one of the models, as if data has migrated from one model to the next. One Protégé user was overheard to say, *"Protégé hard-wires the subclasses into the model!"* We described a similar phenomenon above as a "triple drift".

Again, this can hardly be called a Protégé bug; after all, adding spurious triples into Protégé's files can hardly be counted as normal usage. Or can it? If Protégé were a general-purpose OWL editor, then it is

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

fair game to feed it any OWL files (regardless of their history). But Protégé is a modeling tool, not a general-purpose editor for OWL.

To avoid Protégé issues we have separated BRM-PRM mappings into a model of its own (bridge model). This is a small model developed directly in OWL. It imports both, the BRM and the PRM. Opening this model in Protégé will enable a user to view not only the BRM and the PRM, but also the relationships between them. Modifying this merged model in Protégé, will however cause the problems described above (at least at the time of writing this paper). Any updates to the BRM or the PRM can be made in Protégé using individual models. Any updates to the BRM to PRM should be made directly in the OWL file.

The same approach will need to be taken for the BRM-DRM mapping.

6.3 hasValue reasoning

The ontology design pattern outlined in section 2.4.1 makes use of owl:hasValue. An OWL reasoner can use this to enforce the condition that any instance of a class has a certain value for the specified property. When this property has an inverse, this provides the useful feature that the instance ("Customer Results Measurement Area" in the figure) is linked to all the instances of the class "Customer Results Measurement Category" via the inverse link. Protégé normally fills in inverses automatically when values are given for properties that have them; however, as of build 225, this capability does not work in the case where the value was filled in by owl:hasValue. Unlike the cases above, this is a bug in Protégé, and has been placed on the "to fix" list.

As a result, many items in the PRM and BRM are flagged in red by Protégé. On the occasions in which we filled in the inverse "by hand", Protégé now sees the inverse twice (because of the bug). Many of these properties are functional (i.e., cardinality =1), and hence the double value is flagged as an error. Users who view the models using Protégé may notice these red flags. It is an unfortunate situation, since this makes it difficult to see when Protégé is flagging a genuine error.

6.4 TopBraid®

TopBraid is a tool under development by TopQuadrant. Unlike Protégé, it is not a modeling tool. Neither is it a general OWL processing tool. But it is a semantic web model merging tool, and includes an OWL reasoner. TopBraid accepts RDF files describing OWL modeling constructs, merges them into a single graph, and displays this graph (with OWL reasoning performed over the graph).

An example of TopBraid being used with the PRM is shown below. Unlike Protégé, imports are made in any direction – they are simply graph merges. Also unlike Protégé, reasoning about inverses and hasValue are done by an OWL reasoner. These capabilities (as shown in the screenshot below) allow TopBraid to display something that Protégé cannot:

- TopBraid displays the values known about the Measurement Area "Mission and Business results".

  The left panel shows the classes in context, the center panel all the measurement areas. The right panel shows the values for the selected measurement area. Notice the multiple values for hasMeasurementCategory; none of these values have been specified in the model. All of them were inferred using the hasValue design pattern outlined above, and the combination of hasValue/inverse described earlier.

- TopBraid shows "correct" import of the BRM into the PRM

  The items in the list (LOB_AdministrativeManagement, LOB_CommunityAndSocialServices, etc.) do not come from the PRM, but are instances from the imported BRM. As described in 6.1, such an import goes "the wrong way" for Protégé, but since TopBraid does a general graph import, it is able to support it. The result is that instances that were defined in the BRM as Lines of Business now

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

appear in the PRM as Measurement Categories, as if they had been put there by hand by a diligent modeler, obeying the directive given in the PRM, that the Measurement Categories are the same as the Lines of Business.

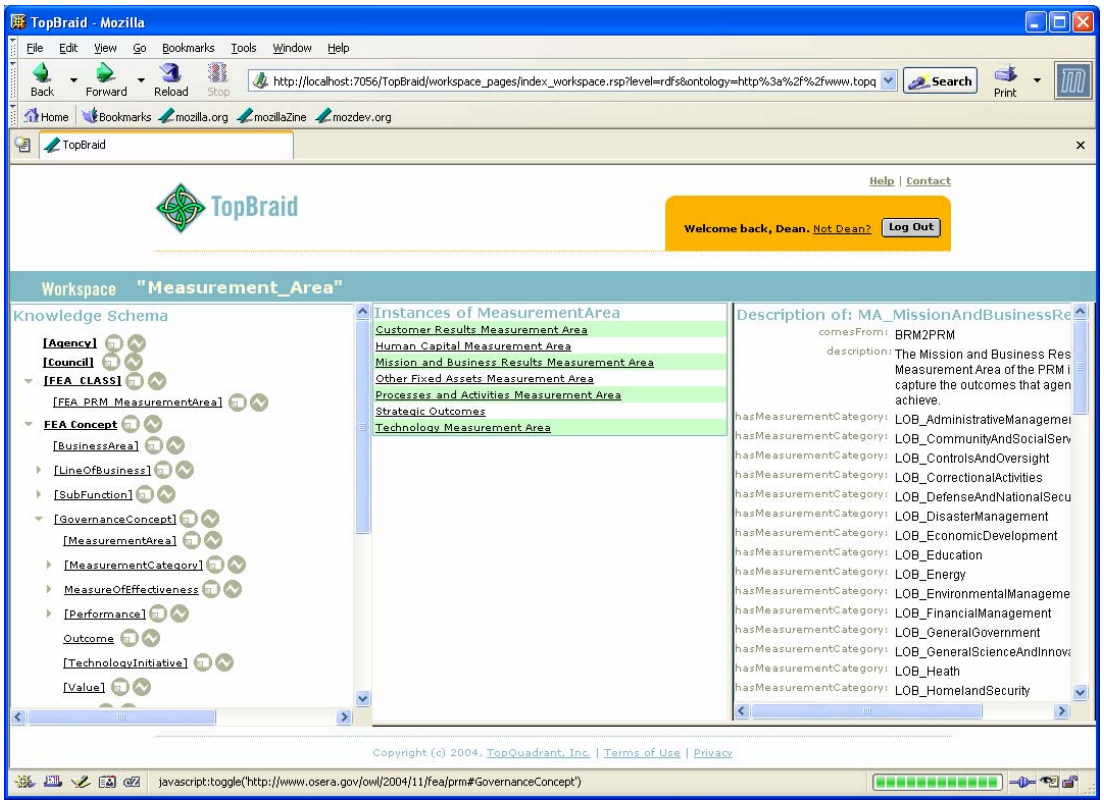


Figure 13: Browsing merged models

6.5 Selected Approach – Different Tools for Different Purposes

Why did TopBraid succeed where Protégé failed? Does it make Protégé obsolete? Not at all.

TopBraid is a different class of tool than Protégé – it is a graph-based tool, based on the foundation of the Semantic Web. It is not a modeling tool, nor is it a general-purpose tool for manipulating OWL files.

TopBraid has very limited capacity for authoring a model. Its capacity for viewing the properties of a model is also limited. For example, while TopBraid can make appropriate use of owl:hasValue and owl:inverseOf, neither of these properties is actually visible through TopBraid; that is, you can't even see why the model behaves the way it does. But it does behave properly, according to the semantics of OWL. The features available (and not available) are by design. TopBraid design goals are different from the design goals of Protégé. They are:

- Provide a way to merge graphs
- Provide a way to view and query merged graphs
- Have a low entry barrier for people who are not expert logical modelers
- Allow an easy way to create OWL files (kind of like MS FrontPage for OWL) using a pre-defined ontology

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

The last design objective is demonstrated in the screenshot below. It shows a user in the Dept. of Energy entering agency specific (operationalized in FEA terms) performance indicators. FEA RMO is used to ensure alignment of these indicators with the PRM generic indicators. TopBraid will save the result as an OWL file.

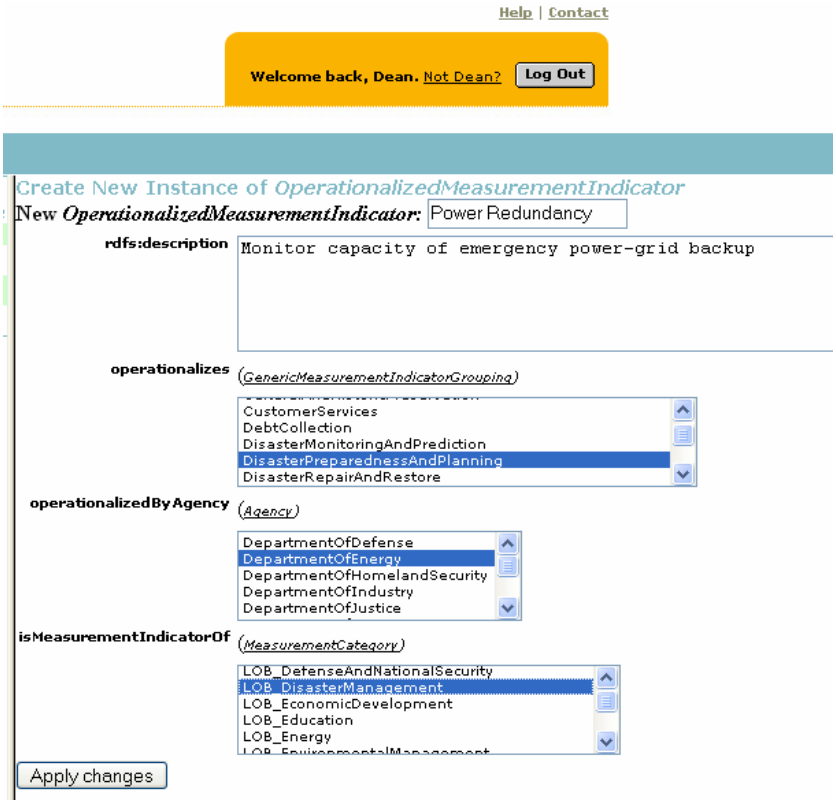


Figure 14: Entering specialized measurement indicator aligned with the PRM

We have designed a way of working using TopBraid and Protégé together, each where it is strongest:

- Experienced OWL modelers use Protégé for editing and viewing the base models. Where Protégé does not deal well with merged models, we factored the merging axioms into *bridge models* (as described in section 5.4), which can be viewed in Protégé but not edited. The bridge models are maintained directly in OWL.
- TopBraid is used for merged viewing for anyone who is interested in FEA RMO.
- It is also used as the basis for further browsing, querying and editing applications, some of which are demonstrated in this white paper.

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

7. Recommendations and Future Plans

This sections describes our "wish list", defining where and how we would like to see the work progressing. GSA is planning to release FEA RMO under an open source license making it possible for anyone to use the models. This raises questions on the best approaches to adoption and governance of FEA RMO. The following sections set out TopQuadrant's ideas and thoughts.

7.1 Incremental Implementation Strategy

Jim Hendler is fond of saying "a little semantics goes a long way." This is the key to how semantic technologies, and the FEA RMO in particular, can gain adoption not only for the formal models themselves, but for the FEA in general. The adoption success will depend on providing small technological and process interventions that incrementally encourage the agencies to move up in the maturity model.

The adoption of the FEA RMO will be an ongoing effort, and will not stop at a simple initial set of use cases described in this white paper. It is essential that new use cases be initiated and followed through as the FEA develops. The process by which this can happen is a sort of ongoing solution envisioning, at each point once new capabilities are available, the next set of capabilities can be considered. In this way, FEA RMO adoption will always be driven by real agency need.

The eventual goal of the FEA reference models is that enterprise architects in various agencies will look at it as a benefit – something that helps them to organize their own enterprise architecture activities, and allows them to produce more streamlined processes more easily. This is a big task for any effort. Fortunately, semantic technologies, like the web itself, allow for an incremental approach that provides a small benefit right away, but is scalable to more value as the network of users grows. The adoption of the technology will move hand-in-hand with the establishment of a community of agencies, in which each participant benefits from the presence of the group.

The incremental strategy outlined in this section served as a guiding principle for specific recommendations that follow. These are presented as recommendations pertaining to governance, technical strategy and deployment options.

7.2 Governance

As for any managed asset, consideration needs to be given to the organization, policies and processes that need to be in place to manage the asset over its life-cycle. Different aspects to this management may be expressed as follows:

1. User Community – enabling adoption
2. Modeling Community – managing evolution of the models
3. Liaison with FEA specifications organization – interfacing with the FEA document authors
4. Version control and release management
5. Feedback management – dealing with comments, issues and enhancement requests
6. Feed-forward management – understanding the effects and desirability of proposed changes

OMB and AIC seemed to be the natural governance bodies for the models. OMB has put the processes in place to evolve FEA document-based models. It would be logical to extend them to encompass FEA RMO.

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

7.3 Technical Strategy

By expressing the FEA RMO in a standard ontology language, we paved the way for a series of applications the provide value to enterprise architects and other information workers in various agencies.

W3C standards represent one of the great advances that has been made in semantic technologies over the past year. By adhering to the standards, systems that provide incremental value (as outlined in the use cases) can be created, without having to lock in to any particular proprietary technology. Standard compliance creates an incremental path for infrastructural technology as well.

This is an important principle, because the state of tools available for working with semantic models is, as for any emerging technology, still in transition. The most widely used tool for semantic modeling is a free software tool provided by a university (Protégé, from Stanford). Systems for reasoning using the OWL standard range from free, university software (Racer) to enterprise level products that could be too costly for pilot projects (Cerebra). This situation will certainly shake out over the next year or two, but FEA needs to start building now.

Our approach to building use cases demonstrations has been to use fast delivery tools. What counts as "fast" depends on the system building resources available; but there is a wide range of inexpensive or free tools available (Jena for Java programmers, RDF Gateway for easy to use scripting languages, Koware for larger scale Java RDF bases, etc.).

7.4 FEA RMO Distribution Options

The FEA models have reached a stage of completion where they reflect the overall structure of the FEA. There are a range of deployment options available at this point. We will outline a few of them below with their pros and cons.

7.4.1 Distribution of OWL files

The easiest way to disseminate the models is to publish the OWL files themselves in their current condition. Consumers of the models will be able to download the files, and process them using any OWL-savvy including but not limited to Protégé.

The main advantage of this dissemination approach is that it is easy to do - the OWL files are already finished, and can be distributed simply by putting links to them on a web page. The only additional work required are instructions for importing them into Protégé[6]. This will allow the broadest variety of possible usage of the files, since the consumers can do with them as they please.

This approach has the following drawbacks:

- Limited audience. The tools that process OWL at the moment are development tools, so the audience for this kind of dissemination consists of developers and consultants with skills in OWL. This is a pretty specialized group.
- We can not guarantee that the models would work with all the possible tools and all versions of the tools. The project scope did not include testing the models with several tools (although we performed some limited testing with SWOOP from the University of Maryland MindLab). And we've already experienced different outcomes with the different releases of Protégé. People having issues loading the models into tools of their choice, may require extensive support and hand holding. Currently there is no organization identified for providing this support.

[6] We single out Protégé for two reasons: (1) currently, this is the most widely used tool; (2) in order to view multiple, inter-related models in Protégé, a user must have correctly set-up policy file – this will require instructions. Alternatively, or in addition to the modular models, a merged model can also be made available. This would eliminate the need for a specially configured policy file.

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

- People who download the models are likely to have questions about modeling choices made. Some the necessary information is covered in this paper, but additional support will be needed.
- The direct usefulness of the models is not obvious.

Our recommendation has been to release the models in the form of files in a controlled way by making them available on request only to a community of interest. Documentation on ontology architecture and patterns included in this paper should be required reading for anyone planning to peruse the files.

We recommend publishing the models using a model browser as described in the next section. We have also experimented with a new Protégé plugin – OWLDoc as a way to document the models. Unfortunately, OWLDoc generated html files do not show rdfs:comments. If extended to include comments, we believe they can provide a useful way to document FEA RMO.

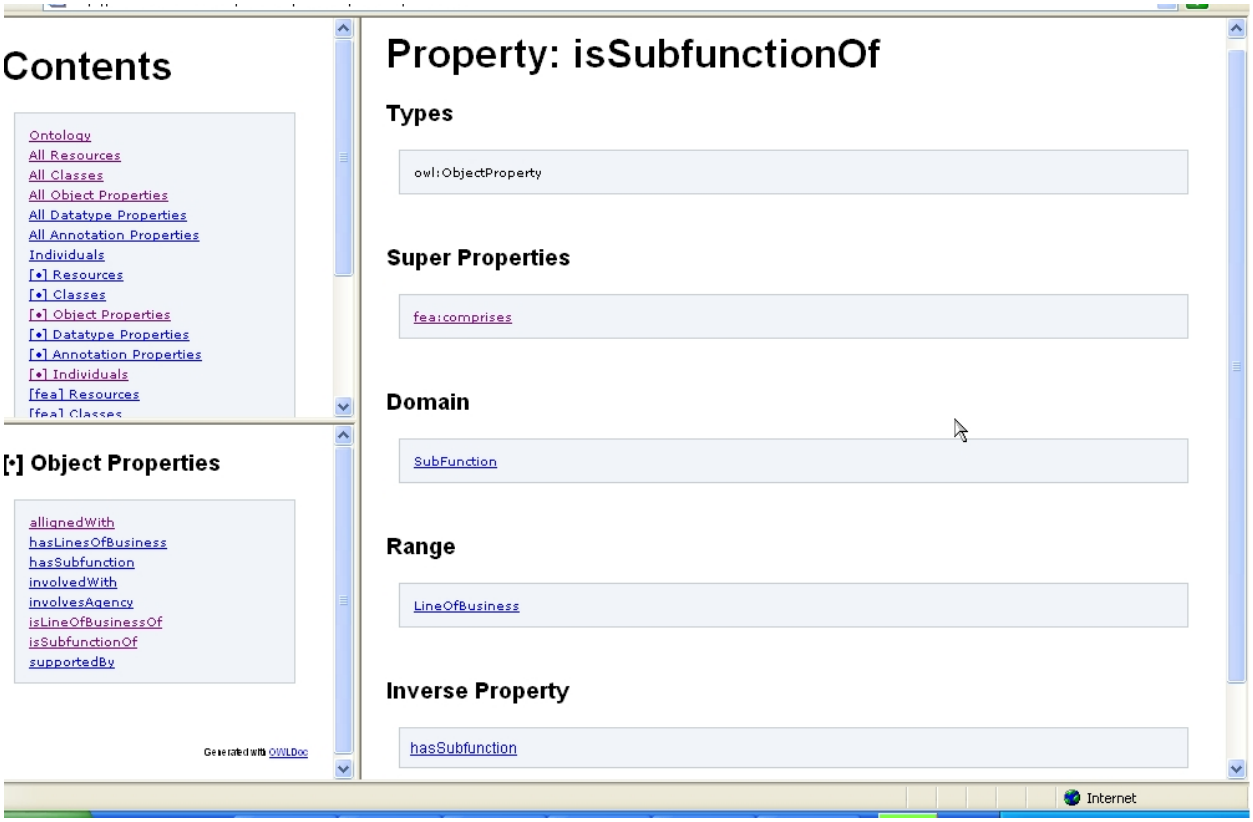


Figure 15: OWLDoc of the BRM

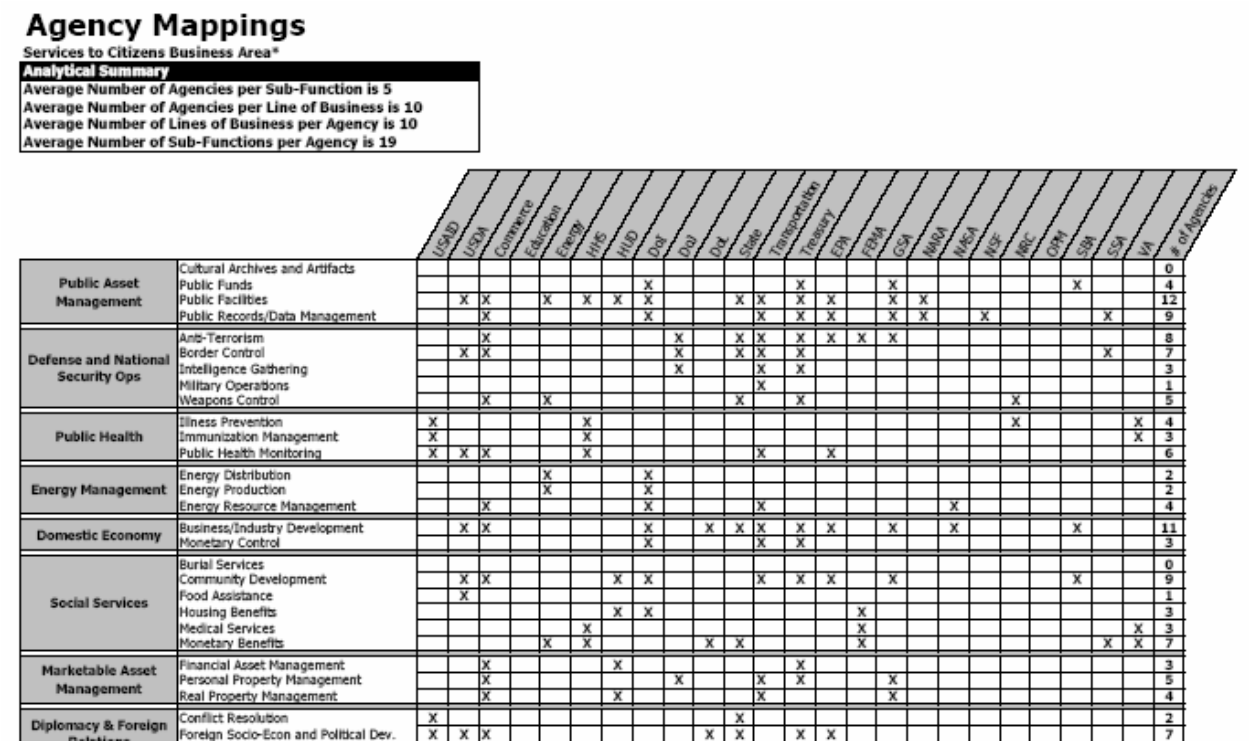
7.4.2 Publication through a model browser

The FEA RMO models can be used to drive a web based "model browser". This can expose the model in more ways than a simple text description. For example, the FEA reference models make considerable reference from one model to the next, and it can be quite confusing at times to understand what the ramifications of such a cross-reference might be. A model like the FEA RMO can be displayed in a number of different ways, bringing in parts of each model as needed to describe a particular aspect.

A model browser can serve as a way to achieve the use cases in the *Semantic Detective* subject area, since it allows for a uniform way of viewing the model. Figures 3, 4 and 5 show currently implemented version of FEA browser. Additional capabilities can be easily added. For example,

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

- References into the original text can be provided from comment and description fields in the models (many of these are already present in the models; others can be developed incrementally to increase the utility of the model).
- A model browser can include an array of creative displays of information from the FEA models. For instance, the following image is a snapshot from the Agency Mappings document of version 1 of the BRM. This diagram shows the lines of business and subfunctions from the BRM, across from the agencies who are active in that subfunction.



Agency Mappings

Services to Citizens Business Area*

Analytical Summary
Average Number of Agencies per Sub-Function is 5
Average Number of Agencies per Line of Business is 10
Average Number of Lines of Business per Agency is 10
Average Number of Sub-Functions per Agency is 19

		USAID	USDA	Commerce	Education	Energy	HHS	HUD	DoI	DoJ	DoL	State	Transportation	Treasury	EPA	FEMA	GSA	NARA	NASA	NSF	NRC	OPM	SBA	SSA	VA	# of Agencies
Public Asset Management	Cultural Archives and Artifacts																									0
	Public Funds								X				X				X						X			4
	Public Facilities		X	X		X	X	X	X			X	X	X	X		X	X								12
	Public Records/Data Management			X					X				X	X	X		X	X		X				X		9
Defense and National Security Ops	Anti-Terrorism			X						X		X	X	X	X	X	X									8
	Border Control		X	X						X		X	X	X										X		7
	Intelligence Gathering									X			X	X												3
	Military Operations												X													1
	Weapons Control			X		X						X		X							X					5
Public Health	Illness Prevention	X					X														X				X	4
	Immunization Management	X					X																		X	3
	Public Health Monitoring	X	X	X			X						X		X											6
Energy Management	Energy Distribution				X				X																	2
	Energy Production				X				X																	2
	Energy Resource Management			X					X				X						X							4
Domestic Economy	Business/Industry Development		X	X					X		X	X	X	X			X		X				X			11
	Monetary Control								X				X	X												3
Social Services	Burial Services																									0
	Community Development		X	X				X	X				X	X	X		X						X			9
	Food Assistance		X																							1
	Housing Benefits							X	X							X										3
	Medical Services					X										X									X	3
	Monetary Benefits				X	X				X	X					X								X	X	7
Marketable Asset Management	Financial Asset Management			X				X						X												3
	Personal Property Management			X						X			X	X			X									5
	Real Property Management			X				X					X				X									4
Diplomacy & Foreign	Conflict Resolution	X										X														2
	Foreign Socio-Econ and Political Dev.	X	X	X							X	X		X	X											7

Figure 16: Agency Mappings

This diagram is nearly two years old – and has not been kept up-to-date. Such a diagram could be automatically generated from the FEA models. In fact, we already have a capability for dynamic generation of cross reference tables in TopBraid.

The particular information in the Agency Mapping table shown above is not available in the current model, but could be added easily enough from the data in the table (especially if we can find the original spreadsheet in which it was collected). The information could then be maintained in a distributed fashion, with various agencies or reporting services updating the information as it comes available. The form will display new rows or columns, as new agency information or new business lines are known.

More dynamic browsing can also be part of such deployment. In the following example from a study done at NASA of the Columbia accident, a complex interplay of Mission Features, Threats and Mitigations is displayed in an interactive display. Using an ontology model, each feature is linked to a number of threats (the number shown next to each one), and each threat has a number of mitigations. Each of these elements is associated with a passage from the Columbia Accident Investigation Report; the selected passages are shown in their new context in the right-hand pane. Similar dynamic context browsing could be made available for the FEA models.

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

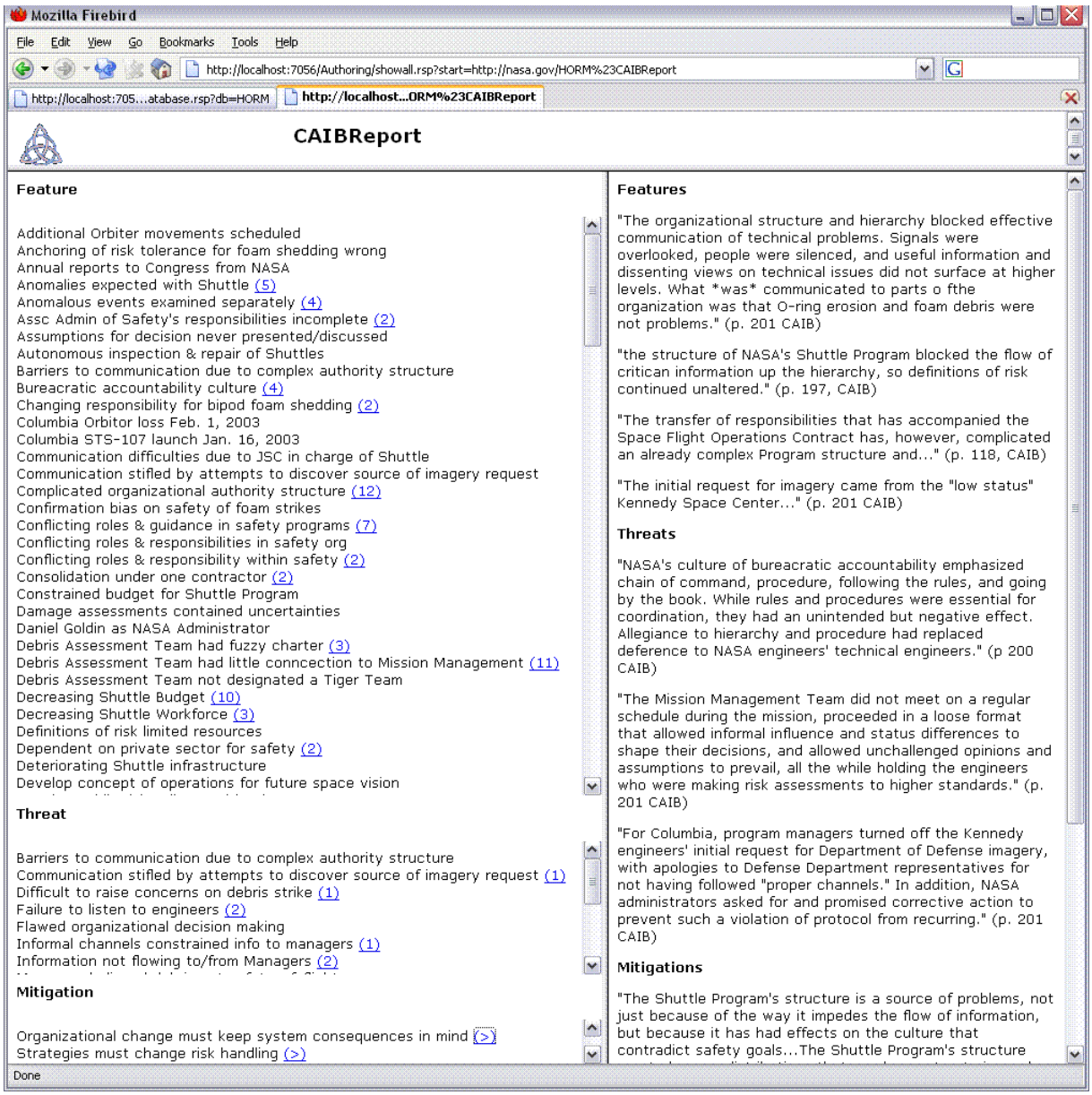


Figure 17: Example of a dynamic ontology-based browser connecting rich content

The advantage of browsers of this sort is that they:

- Make the FEA models more accessible to a wide range of users.
- Can also merge data from multiple sources, into a single browsable unit, allowing the maintenance and use of the FEA models to be distributed among agencies, thereby providing an adoption path to more widespread use of the FEA.

There are no apparent drawbacks to this approach. It has been easy to implement and we feel that the implementation can be extended in interesting ways. Simple applications of this kind (with appropriate adjustment from someone who has a good grasp of the needs of the agencies who are in the audience of the FEA) will go a long way toward demonstrating the viability and utility of the models. Such browser

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant

can be enhanced with “community” features. By this we mean, ability to comment on and discuss different aspects of the FEA models.

7.4.3 Reference Implementation

A reference implementation for FEA RMO using HP Jena version 2.1 can be made available. This is a short Java program with 4 functions:

- Loading a model
- Traversing the model
- Loading reasoners
- Querying the model

The program can be used as a reference example for parties wanting to build applications based on the FEA RMO.

Copyright ® 2004-2005 TopQuadrant, Inc.
All Rights Reserved. Printed in U.S.A. Confidential, Unpublished Property of TopQuadrant