Utilizing Metadata as a Knowledge Communication Tool

R. Todd Stephens
Nova Southeastern University
todd@rtodd.com

Abstract

The modern enterprise environment is a collection of technical, functional, application, and data related assets loosely connected within a heterogeneous environment. Communicating the inventory, meaning, and eventual understanding of the knowledge held within the corporation can be a daunting task. This paper presents an detailed methodology and case study of building enterprise business intelligence from a multi-dimensional metadata framework. The framework includes structural, integration, and semantic metadata as well as the physical components required to deliver the functionality to the end user. The case study section will review a large telecommunications effort at communicating the knowledge of enterprise business intelligence.

Keywords: metadata, enterprise architecture, asset management, knowledge management.

1. Introduction

Enterprise metadata and business intelligence come together to provide one of the most powerful communication tools within the organization. The rationale of enterprise business intelligence is to discover the utilization of capital assets within the organization. The current level of technology can deliver knowledge to the end user but more advanced methods are needed in the areas of taxonomies, ontologies, standards, and automation. Most researchers will agree that a broadly-based approach is needed, including consideration of how users understand, navigate, and communicate knowledge embodied in computer-based vocabularies and metadata classification schemes.

Implementations of structured metadata solutions must overcome a variety of problems including: mixed vocabularies, content and structure of the meta-model, the variety of asset structures, and integration of less structured knowledge placed in documents, business processes, and web pages.

1.1. Enterprise Asset Collection

The enterprise architectures define a universe where assets are created by the technical community in a variety of forms (Layer 1). An asset is any person, place or thing within the technological community. Examples of assets include: databases, logical models, physical models, XML structures, components, documents, metrics, systems, interfaces, etc. A resource would be similar to an asset with the exception that resources come from outside of the organizational walls. Resources could include research services, web services, packaged models, etc. Enterprise Business Intelligence (EBI) is built upon this foundation of assets. Figure 1 presents a layered view of the role enterprise metadata plays in the communication of knowledge via enterprise business intelligence.

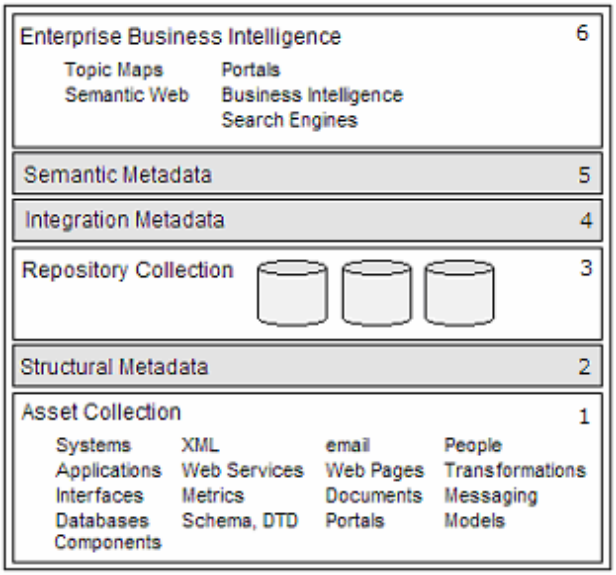

Figure 1. Enterprise Metadata Framework.

2. Structural Metadata

Metadata has traditionally been defined as “data about data” or “information about information”. Pöyry, Pelto-Aho, and Juha Puustjärvi [1] define metadata as a

discipline that is descriptive and classifying information about an object. Metadata describes data, information, and knowledge [2]. Today, with the advent of technologies such as hypermedia and heuristically-based searching and indexing, a new, broader, more generic definition of metadata is needed. This definition should include the traditional concepts, but it should add the concepts of existence, perspective, modeling, and topicality. A new definition should recognize that much, if not most, of enterprise data is not found in traditional Relational Database Management Systems (RDBMS), but rather, it is found in the myriad technological assets and views of those assets that exist at any point in time. The enterprise definition of metadata is as follows: "Metadata is structured, semi-structured, and unstructured data which describes the characteristics of a resource (external source) or asset (internal source). Metadata is about knowledge, which is the ability to turn information and data into effective action."

Metadata can provide abundant information about where an asset is located, what primitive elements make up the asset, how the asset was developed or created, where the asset is physically located, who the steward of the asset is, and, of course, an inventory of what assets exists. Scientists, researchers, and business practitioners continue to redefine, re-scope, and re-purpose the basic utility of metadata. All of this utility is a result of the underlying meta-model that holds the information. Does this utility contain the relationship between assets? Building a semantic understanding of assets is also a critical function of metadata. When describing an asset there are two different levels in which the asset can be described and eventually loaded into a meta-model. Meta-model is a term used to describe the logical and physical model for a database that holds metadata information accessed by a repository. The meta-models in our framework fall into two categories: generic and context specific meta-models.

The generic meta-model is simple and straightforward metadata such as name, description, or keywords. The Dublin Core standard is one such generic meta-model standard (Integration Metadata, section 4.0). The context specific meta-model describes a specific asset within a specific context: the Object Management Group (OMG), Common Warehouse Model (CWM), Corba, and Web Service Definition Language (WSDL). These standards focus on specific types of resources or assets (Structural Metadata).

Structural metadata (Layer 2) is the category where the context specific meta-model information will be stored. Each type of asset as well as the context of the asset will have different data collection points. Static databases will have different metadata components than the logical view of the data which will be different than the transformation of that data from one source to another.

At the heart of any repository is the core constructs that make up the meta-model. The meta-model is simply the elements that are used to describe the asset. The same principles that exist for a entity-relationship model apply to a meta-model as well. While we indicate ER diagramming as the technique, any modeling type language could work: UML, Class-Objects, etc. For example, the following elements can be used to describe a table asset:

Table 1. Basic Metadata Components for Data.

Name	Description
Table Name	The name of the table
Database Name	The name of the parent database
Keywords	Specific keywords and phrases about the table
Description	Detailed description of tables utility
Date	Date the information was published
Schema	Related Source Schema
Logical Model	Parent logical model
Owner	Owner of SME of table
Server	Physical location of table

While meta-models vary from implementation to implementation, we can categorize them into four distinct models: proprietary models, standards based models, functional or industry models, and requirements based models.

Just about every tool out there has an underlying physical meta-model which requires a proprietary model integration. These mete-models are designed to work with the specific tool. Logical or Unified Modeling Language (UML) models for these tools are much harder to get access too. The real question here is the model open, extensible, and open to metadata interchange.

There has been a ton of work at trying to create a standards based meta-model. ISO 11179, CWM, and Metadata Object Facility (MOF) are all efforts at creating meta-models that allow meta-exchange through a collection of standards. Developing standards is a response to the enterprise need to integrate valuable data spread across organizations from multiple sources.

More and more standard schemas are being developed in order to share information across and between the corporations. For example, XML.org houses a centralized repository described by the following: The XML.org Registry is a community resource for accessing the fast-growing body of XML specifications being developed for vertical industries and horizontal applications. The XML.org Registry offers a central clearinghouse for developers and standards bodies to publicly submit, publish and exchange XML schemas, vocabularies and related documents. Operated by OASIS - the non-profit XML interoperability consortium -- the XML.org Registry is a self-supporting resource created by and for the community at large. Industry and

functional meta-models are part of this movement although an XML representation is not required.

The final model is one that is developed in house and is proprietary to the organization itself. These models are driven by the technical and business requirements that deliver a flavor of uniqueness into the metadata strategy. These models provide the basic utility required by the structural metadata.

3. Repository Collection

The main reason employees come to the repository (Layer 3) is locate and gather information. The content of a repository is not limited to the metadata content provided. Rather, content includes the solutions and strategies employed to make it easy for the user to accomplish important tasks, such as information retrieval, search, and navigation required in creating knowledge feedback [3]. Becker and Mottay [4] define information content to include timely and correct error messages, prompts, button labels, textual description, help, and customer service information. For a global perspective, repository designers should be careful not to lose specific meaning in the translation or the use of specific symbols such as the shopping cart. The repository site gives an organization the ability to present almost limitless information on their assets. This information or content should include the metadata and service quantity, quality, and relevance to the customer [5].

4. Integration Metadata

While structural metadata focused on the asset specific information, integration metadata (Layer 4) focuses on the generic information that can describe each asset. Information elements like name, description, type, and author are generic in nature and could describe any asset in the enterprise. The current web environment can support a single threaded keyword search of textual information. Unfortunately, the volume of information is making this model more and more unusable [6]. The efforts under the Semantic Web umbrella are working toward providing a better method of searching the web via the use of vocabularies like the Dublin Core framework. This framework describes the schema of metadata that can be embedded in Hypertext Markup Language (HTML) or Resource Definition Framework (RDF) [7].

The key to providing semantic knowledge is an agreement on the standards of documentation. The Dublin Core Metadata Initiative (DCMI) is an organization that is working on creating a standard set of descriptive tags through an open source type organization. The standard summarizes the updated definitions for the Dublin Core metadata elements as originally defined by the DCMI. These definitions are officially known as Version 1.1. The definitions utilize a formal standard for the description of metadata elements. This formalization helps to improve consistency with other metadata communities and enhances the clarity, scope, and internal consistency of the Dublin Core metadata element definitions [8]. There are 15 basic elements defined by the DCMI, which are presented in Table 2.

Table 2. Dublin Core Metadata Set.

Element	Description
Title	A name given to the resource
Creator	An entity primarily responsible for making the content of the resource
Subject	The topic of the content of the resource
Description	An account of the content of the resource
Publisher	An entity responsible for making the resource available
Contributor	An entity responsible for making contributions to the content of the resource
Date	A date associated with an event in the lifecycle of the resource
Type	The nature or genre of the content of the resource
Format	The physical or digital manifestation of the resource
Identifier	An unambiguous reference to the resource within a given context
Source	A reference to a resource from which the present resource is derived
Language	A language of the intellectual content of the resource
Relation	A reference to a related resource
Coverage	The extent or scope of the content of the resource
Rights	Information about rights held in and over the resource

Each of these elements can provide vital information pertaining to the usage, purpose, content and structure of the web page or any other web-based object. Some of these elements are broken down into further qualifications such as the “Date” element. The qualifiers of the “Date” element include valid date, issued date, modified date, available date, and created date. These qualifiers provide additional semantic definitions that enable a closer definition of the semantic meaning of the object.

5. Semantic Metadata

Semantic metadata (Layer 5) is the assembly of assets based on explicit or implicit elements. Explicit assembles could include hierarchal structures, such as the ones found within search engines like Yahoo. Implicit assemblies could include inference engines that traverse the corporation looking for relevant assets. Keyword base search engines provide a basic example of how this process might work in the future. Currently, a user types in a few keywords and the search engine returns a set of

documents that contain some or all of the keywords. This functionality should be expanded to include any asset in the corporation and not just the ones documented in a web page or document. In the future, agents will be able to traverse operating systems, XML constructs, interfaces, and other asset that can viewed by the computer system. Assets will then be able to be grouped by context, usage, time, and various other constructs.

5.1. Search Utility

Locating assets within the repository is a critical function of the repository. Repository lore tells us that every piece of content should take no more than three clicks to access. The "three-click" rule has been around for some time now but is it valid and more importantly does it have validity in the metadata world. While this theory has been disproved by the User Interface Engineering [9], they do point out the level of dissatisfaction decreases as the number of clicks increase. There are three basic problems that need to be addressed within the search utility environment. First, the search engine must properly specify the information required for the initial query. This information must be extracted from the user in a usable fashion. The second is finding items relevant to the information specified. Finally, the user must be able to judge the quality of information provided by the engine.

There are two basic search utilities that should be included in the repository. The first is the internal repository search utility that will search within the asset collection and return the list of assets to the user. This allows the user to stay within the context of the repository in order to gather information about the particular asset. The second search engine moves outside the repository and searches the entire collection of assets regardless of the type of integration or repository architecture.

Building a search result and request function is simple when compared to building the engine. Regardless of what the meta-model looks like, there are a few design principles that should be implemented [1].

1. While the average size search phrase is 2.2 words, providing an advanced search is a wise decision. The simple search should include a text box that will search the key structures within the meta-model. The advanced search should include options for all of the major elements. Nielsen (2000) describes the importance of locating the search utility on every page as an imperative to a successful repository. Users that primarily use the search utility will need to be able to locate it on the home page as well as any other page within the site.
2. Allow the user community to focus on the search results by removing much of the noise that may exist in the repository. Much in the same way most retailers remove the options from the user view once checkout has started.
3. Easy to use list is the preferred format for result presentation.
4. Place the key elements of the meta-model on the list. Items like asset name, type, sub component count, last update, etc.
5. Display the customers search request. (i.e. "You searched for 'Customer Elements')
6. Include the search box within the results, this allows the user to perform a new search without linking back to the original search page.
7. Be sure to include object count and perhaps the current sort order. (i.e. 234 Reults Found, Sorted by Name)

6. Enterprise Business Intelligence

Enterprise Business Intelligence (Layer 6) is about the dynamic assembly of knowledge about the organization, business environment, competitive environment, and resources available to the organization.

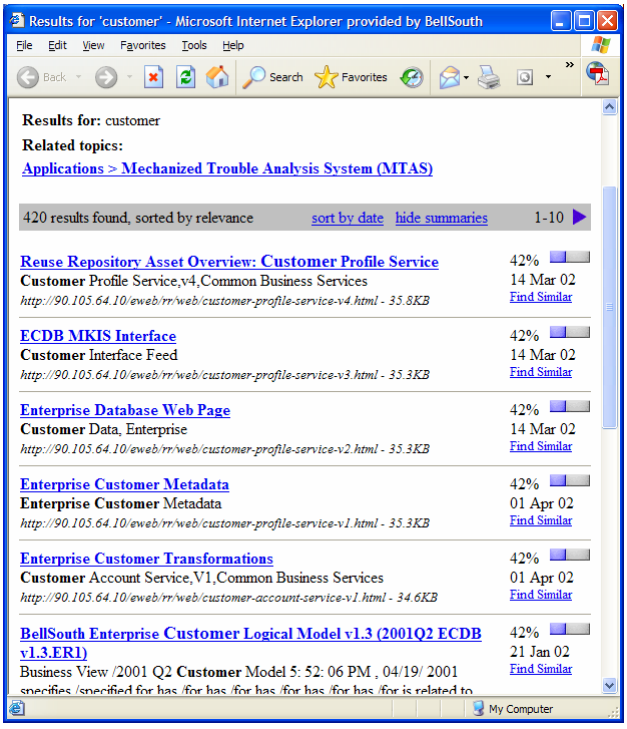

Figure 1. Search Results.

6.1. Basic Category and List Conditions

Figure 1 provides a search engine result set where the collection includes all of the technology community assets assembled with integration metadata. The result set is based on the search term of "customer" which included 420 assets. Each of the assets includes a title, description, repository link, asset date, and a score variable. The list view provides very little context information of the particular asset being presented. The addition of asset summaries provided contextual disambiguation at the price of presenting less results per page. [11].

The complexity of the result set is not in the formatting of the search results but rather in the assets presented. Normal search return unstructured content held within document objects and web pages. With structural metadata, we are adding a new dynamic to the knowledge communication paradigm.

6.2. Structural Metadata Viewer

From the search result set, the end user could be placed directly in the repository viewing the asset of representation. Figure 2 provides a service asset description found within the service repository.

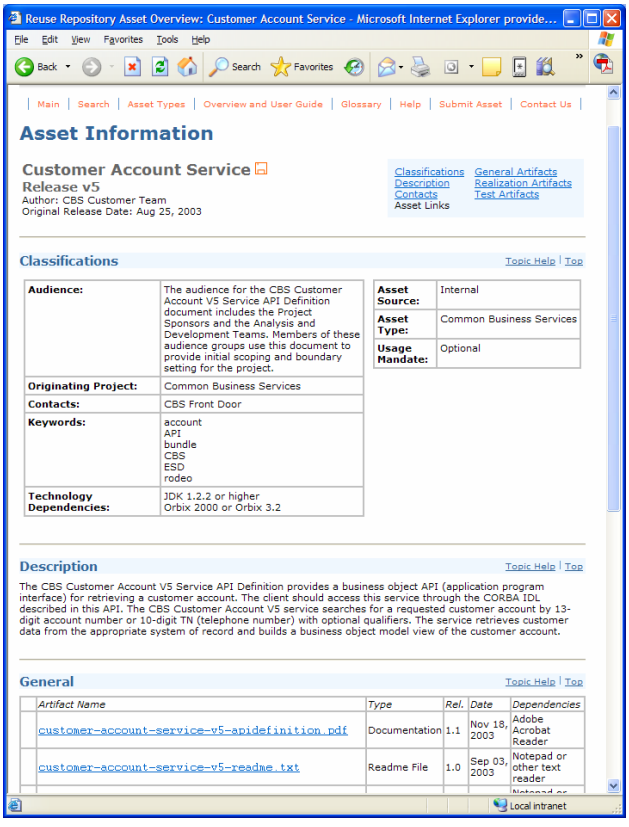

Figure 2. Repository View.

In this repository, the user can view the detailed structured metadata that describes the customer account service. The detailed metadata includes name of the service, date of creation, release, audience, contacts, description, and associated documentation attached to the service. This information includes the installation guide, installation package, and test cases.

7. Discussion

The results of the implementation provided valuable insight for metadata professionals and knowledge management specialist. Detailed statistics have been captured around the content and usage of the repository collection for a period of 24 months. During this period detailed inventory statistics were collected by the utilization of an object counter. The usage statistics were collected by using the NetIQ's WebTrends product. The following observations indicate the impact of adding the assets to enterprise environment.

7.1. Increased Usage

During the review period, the average increase in usage or page views within the repository collection was 27.8%. The amount of time spent within the repository also increased an average of 5.12%. Users that spend more time and visit more often increase the overall communication value of the asset collection.

7.2. Increased Repositories

The original repository collection contained five repositories that housed database, data movement, systems, interfaces, and component metadata. Over the past 14 months four additional repositories have been added to the collection to view web services, open source, XML artifacts, and metrics. The new repositories account for 35% of the increase in assets housed in the collection.

7.3. Increased Assets

The total number of assets increased from 132,000 to 153,000 during the period of review. While new repositories account for a percentage of the increase, the vast majority of assets were accounted for in the systems/interfaces, database, and components.

7.3. Increased Reuse of Assets

Reuse of service based assets grew 87% during the period of review. The primary indicator of the amount of reuse in a product or application is based on the percent of the unit of measure of reuse versus the entire application calculation. The reuse percent result indicates

that for every line of code written within the application, 50.16% is reusable. However, the percent reuse level does not necessarily reflect the effort that was saved from reuse. The Reuse Value Add (RVA) calculation estimates the contribution of the reusable asset. This calculation reflects positively on an organization that creates reusable assets as well as helping others create reusable assets. The following calculation is based on a development asset being measured by the LOC. The calculation is pretty straight forward resulting in a Reuse Value Add of 199.34%. This means that for each line of code we add 99.34% additional value is added to the organization

8. Future Work

Adding structured assets to the repository collection as well as the search engine results has improved the overall communication and understanding of what assets we have within the corporation. This paper reviewed the impact from an organizational perspective and not from the end user perspective. While there doesn't seem to be any negative effects of capturing structured information more studies should review the impact from the user perspective. Additional work should be done in the assimilation of this knowledge and provide additional methods beyond the search engine and repository. Finally, human computer interaction studies could take a deeper look into the usability of a repository and find methods to present multiple assets in a single collection or view.

10. References

[1] S. Bowman and C. Willis. *Design ways: Designing web sites that sell*. Rockport Publishers, Berkley, CA, 2002.

[2] A. Tannenbaum. Metadata Solutions: *Using metamodels, repositories, xml, and enterprise portals to generate information on demand*. Addison Wesley Professional, Boston, MA, 2001.

[3] C. Calongne. Designing for website usability. *Journal of Computing in Small Colleges*, 16(3), 39-45, 2001.

[4] S. Becker and F. Mottay. A global perspective on website usability. *IEEE Software*, 18(1), 61-54, 2001.

[5] J. Palmer. Designing for website usability. Computer, 35(7), 102-103, 2002.

[6] S. Decker, P. Mitra, and S. Melnik: Framework for the semantic web: An RDF tutorial. *IEEE Internet Computing* 4(6): 68-73, 2000.

[7] O. Lassila. Web metadata: A matter of semantics. *IEEE Internet Computing*, 30-7, 1998.

[8] Dublin Core Online Site http://www.dublincore.org/

[9] User Interface Engineering Online Site http://www.uie.com/

[10] J. Nielsen. *Designing web usability*. New Riders Publishing , Indianapolis, IN, 2000.

[11] S. T. Dumais, E. Cutrell and H. Chen. Bringing order to the web: Optimizing search by showing results in context. *Proceedings of CHI'01, Human Factors in Computing Systems*, April, pp. 277-283, 2001.

About the Author

Dr. Todd Stephens is the technical director for the Metadata Services Group for a major telecommunications company. Todd is one of the foremost authorities on the study of structure with over 70 academic and professional publications as well as nine patent pending applications. Todd holds degrees in Mathematics and Computer Science (B.S.), and Information Systems (MBA, Ph.D.). Todd is a member of IEEE, ACM, DC-Corporate, Upsilon Pi Epsilon, and DAMA International.