A BPTrends Column Updated Feb. 2013

MDA Journal

David S. Frankel
David Frankel Consulting

david@dFrankelConsulting.com
http://www.linkedin.com/in/davidsfrankel

Author:
Model Driven Architecture:
Applying MDA to Enterprise Computing

Semantic Metadata for Data Integration

Making Semantic Vocabularies and Ontologies Actionable

I've written extensively in previous MDA Journal articles[1] about how semantic interoperability problems drive up integration costs across industry, and have outlined some approaches to getting a handle on the problem. In an economy where a company's business network of suppliers, distributors, partners, and customers is an increasingly important source of competitive advantage, semantic interoperability – the ability of human and automated agents to coordinate based on a shared understanding of the data that flows among them – is a major economic enabler.

In this Column, I'd like to explain where semantic ontologies fit into this picture.

[1] See the following articles:

1. "Industrial Convergence," MDA Journal, BPTrends, July 2007.
2. "Semantic Interoperability and Convergence in the Financial Industry," MDA Journal, BPTrends, October 2007
3. "Semantic Interoperability Roadmap," MDA Journal, BPTrends, February 2008.
4. "XBRL and Semantic Interoperability Roadmap," MDA Journal, BPTrends, March 2009

The Power and Limitation of Semantic Ontologies

Discussion of semantic interoperability problems tends to attract advocates of the Semantic Web, who have a lot to offer. They point out that they can use Semantic Web languages to build ontologies of the fundamental concepts in a given domain. Semantic Web experts have built ontologies that they believe can be useful in business domains such as banking, financial reporting, health care, and so on, and I agree with their assessment.

The question that we must be able to answer is: exactly how can we put semantic ontologies to use? At a recent gathering where people were discussing semantic interoperability issues in financial lines of business, a Semantic Web expert showed an ontology of financial concepts that he and some others had built, which was indeed fine work. At the end of his presentation, a representative of a major bank asked what the relationship is between that semantic ontology and the concrete data structures such as relational databases, XML schemas, XBRL taxonomies, and application data stores that his bank has to manage and map to each other. The questioner, a very sophisticated senior architect who understands the semantic interoperability problem, had a hard time understanding how to use the ontology to address that problem, even though the ontology's clean specification of semantic concepts had a certain appeal.

My takeaway from this anecdote is that it isn't enough to build semantic ontologies without addressing a basic point: What is the link between the semantic ontologies on the one hand and the concrete data structures that we have to deal with? Without addressing this matter, we cannot put the ontologies to work toward mitigating the semantic interoperability problem.

Data Integration – How Far Have We Advanced?

To understand the problem that semantic metadata addresses, it's useful to step back and assess the current state of data integration in enterprise IT.

The Integration Analyst's Pain Point

An integration analyst takes on the task of mapping one concrete data structure to another, which has to be done all too frequently when integrating systems. The structures that the analyst must map are often lengthy and complicated, containing many data elements.

Current state-of-the-art data mapping tools lack the ability to help the analyst figure out what the mapping should be. They display both data structures on the screen and allow the analyst to graphically draw connections and write expressions to specify what should map to what according to what rules (see Figure 1, where Sales Order 1 and Sales Order 2 are two different sales order formats). These tools take the graphical map as input and produce an executable transformation, in keeping with a model-driven approach that is a genuine advance over having to write transformation programs in lower-level code as per the predominant practice of a decade ago. Thus, once the analyst has figured out what the mapping should be and has entered the mapping into the tool, the tool does useful things.

However, it is very time consuming and error prone for the analyst to figure out what should map to what. Consequently, subtle mistakes occur in data transformations used to integrate systems, costing the involved parties serious money. Even if the analyst has access to good documentation of the message formats, the size and complexity of the formats means the process is fraught with opportunities for mistakes.

In sum, once the analyst has decided what should map to what, current-generation data integration tools simplify the mechanics of describing the mapping and getting it into executable form, and that is really a big help, but it's not sufficient, because if the analyst makes the wrong decision the tools will simply make it possible to execute the wrong decision quickly. This is the essence of the semantic interoperability problem.

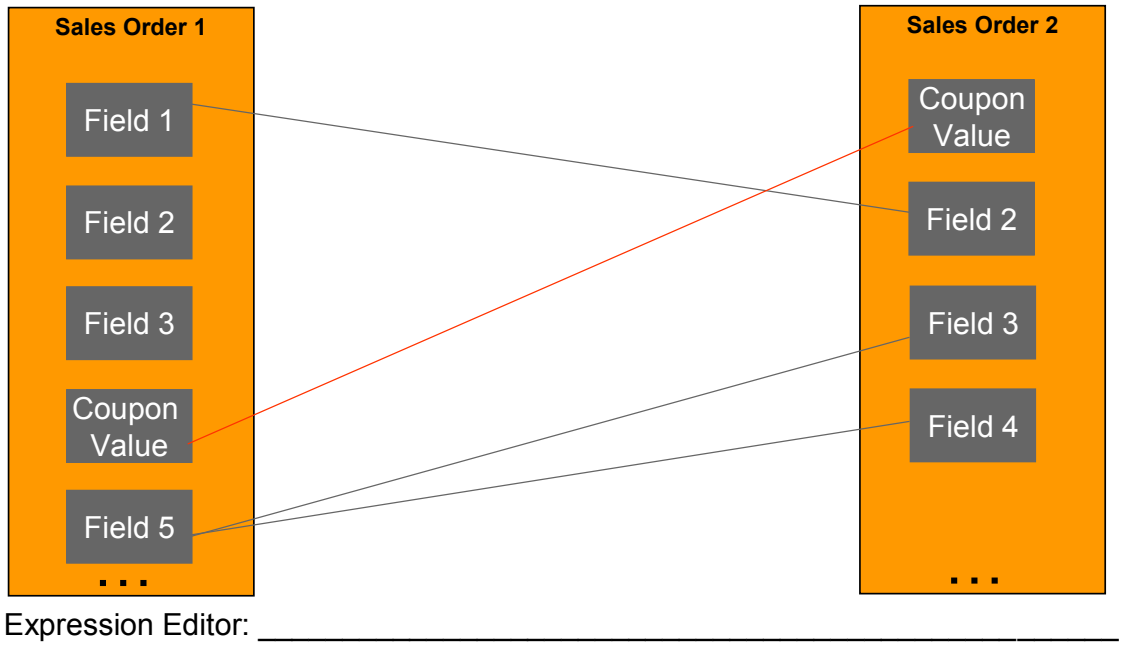

Figure 1: A Current-Generation Data Integration Tool

How Tractable is the Semantic Interoperability Problem?

Industry practitioners whom I work with generally agree that the advance in integration tooling of the past decade whereby the model-driven approach now predominates has addressed about 20 percent of the data integration bottleneck, with semantic interoperability limitations accounting for the remaining 80 percent of the problem. The 20 percent improvement in integration analysts' productivity and accuracy is truly a substantial achievement. The problem is so large, that modest advances have a big economic impact.

Studies indicate that another improvement of similarly modest magnitude could add measurably to the global GDP.[2] Even if such projections are too optimistic, it should still be evident from the enormous costs of integration that moderate gains in semantic interoperability could produce substantial returns.

Thus, in attacking this problem, we do not have to achieve fully automated semantic interoperability. Complete automation of all mapping decisions is probably not attainable, certainly not in the foreseeable future. Humans will have to be involved in the mapping decisions and must be able to override machine-generated suggestions. But we can do better than we are doing today.

Semantic Metadata –Linking Semantic Ontologies to Concrete Data Structures

As I've written in the previous articles, a number of standards initiatives -- and major ERP vendors to some extent – are attacking the semantic interoperability problem using techniques based on the ISO 11179 and UN/CEFACT Core Components standards. Their techniques are closely related to the subject of semantic metadata.

[2] Joseph N. Bugajski, *Response to Payments RFI*, Visa International Payments Association, August 22, 2004, OMG document finance/04-08-07.

The Basics of Semantic Metadata

In order to explain what semantic metadata is, I begin by explaining a few elementary principles of human language, which I will then use to describe the basic tenets of semantic metadata.

Figure 2 illustrates the following:

1. We have a *dictionary* for a language (such as English), which contains words and the definitions thereof.
2. We have a *thesaurus*, which defines relationships among words such as homonyms and synonyms
3. Words play specific roles in sentences, the roles being defined by the rules of grammar. Those roles include *subject*, *verb*, *indirect object*, *article*, *direct object*, and so on. In terms of Figure 2, note that the definition of "food" in the dictionary would not specify that "food" is a direct object, because it is not inherently a direct object. In another sentence "food" could be the indirect object.

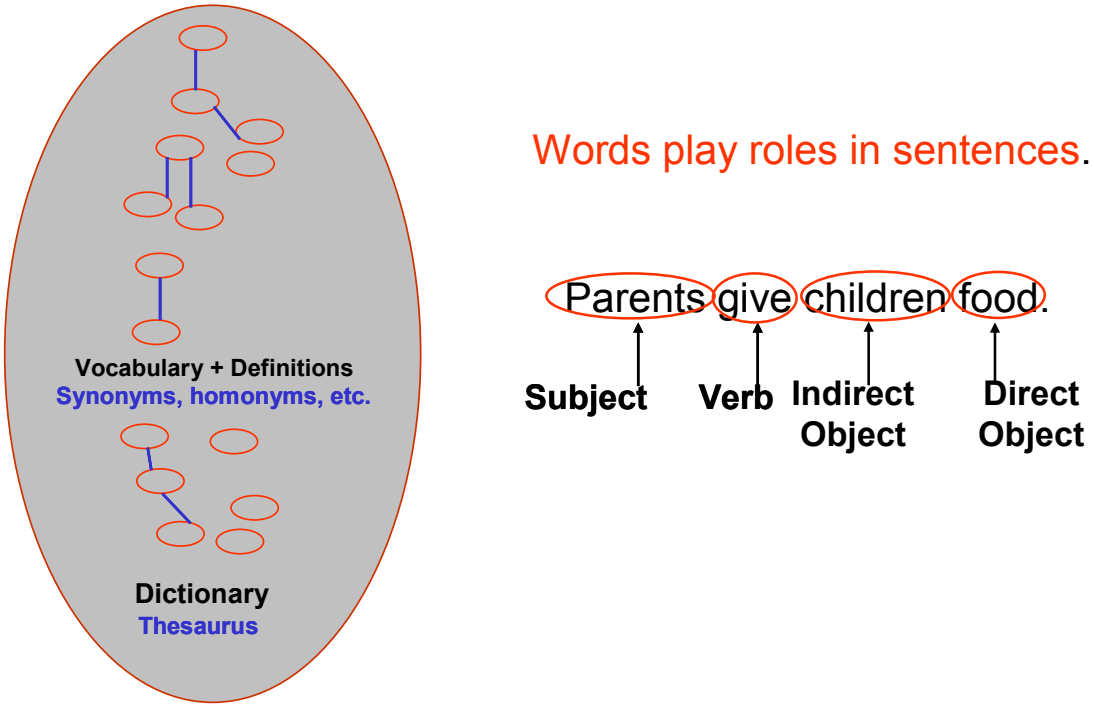

Figure 2: Some Basic Elements of Human Language

Just as the rules of English grammar define the roles that the dictionary's words play in a sentence, the rules of semantic grammar define the roles that elements of a *controlled vocabulary*[3] play in a structured data element. Figure 3 illustrates the following:

1. We have a machine-readable controlled vocabulary. The controlled vocabulary is analogous to the dictionary of a human language.
2. Thesaurus-like relationships among the elements of the controlled vocabulary are also encoded in machine-readable form.[4]

[3] A controlled vocabulary is a compendium of terms that is governed by some authority,.

3. Semantic grammar rules define specific roles that elements of the controlled vocabulary play in structured data elements, those roles being *property term*, *property qualifier*, *representation term*, *object class term*, and *object class qualifier*. Informally we can think of these rules as constituting the grammar of data elements.

4. The controlled vocabulary and thesaurus can be made machine-readable via languages such as the Ontology Web Language (OWL) and the Resource Description Framework (RDF), which are two related pillars of the Semantic Web.

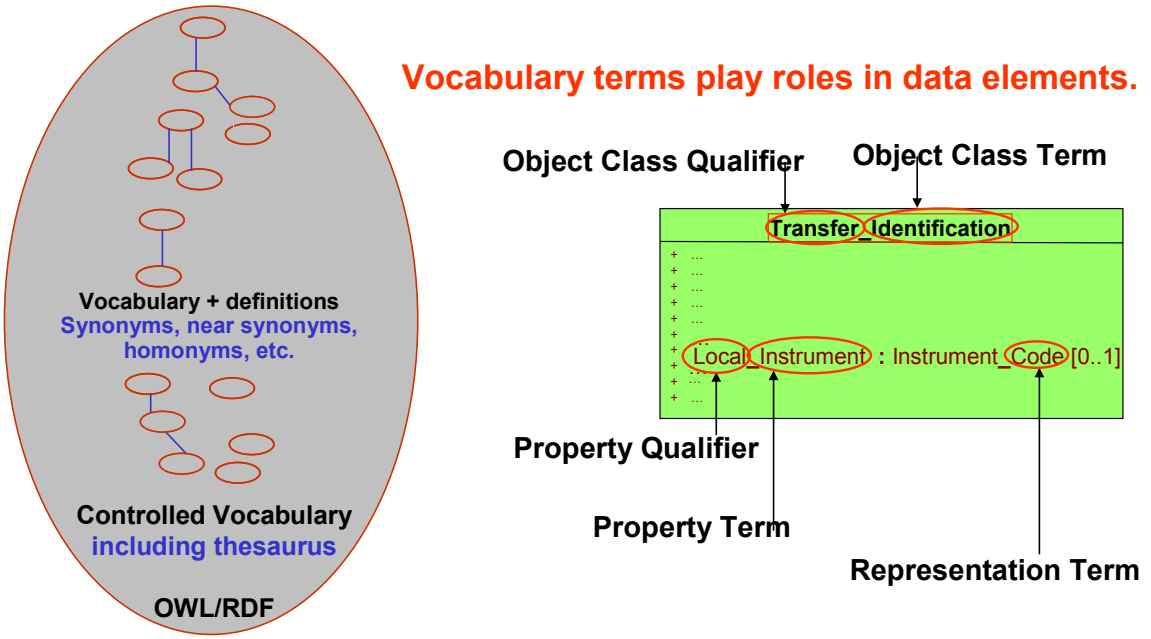

Figure 3: Semantic Metadata – Linking Data Structures to Ontologies

In Figure 3, the semantic metadata specifies which concepts from the controlled vocabulary play which specific roles in the structured data element 'Local_Instrument."[5] For example, the designation that "Instrument" (an element of the controlled vocabulary) plays the role of property term is a piece of semantic metadata. Another piece of semantic metadata is the designation that "Local" (an element of the controlled vocabulary) plays the role of property qualifier. Semantic metadata thus links the concrete data structure to the ontology that encodes the controlled vocabulary.

Let's analyze the roles that elements of the controlled vocabulary play in data elements:

- *Object Class Term*: The overarching concept that ties together a set of data elements, which is "Identification" in our example.
- *Object Class Qualifier*: Qualifies the Object Class Term, such as "Transfer" in our example. There can be multiple Object Class Qualifiers.
- *Property Term*: A charcteristic of an object, such as "Instrument" in our example.
- *Property Qualifier*: Further qualifies the property term. There can be more than one property qualifier per data element.
- *Representation Term*: A very high-level data type such as "Code" in our example. The Representation Term makes no commitment as to whether the code is implemented as a

[4] In ISO 11179, the thesaurus is actually part of the controlled vocabulary

[5] In the diagram, the data type "Instrument_Code" follows the data element name "Local_Instrument" and a colon, as per UML notation. This is not to say that the data structure is necessarily a UML element – I am just using UML notation for presentation purposes.

number, an alphanumeric, an enumerated type, or whatever; it simply says that, semantically, this element is a code. UN/CEFACT has defined 22 representation terms that it calls *Core Data Types*.[6]

Note that an atomic term in the controlled vocabulary can consist of more than one word. For example, the authority that maintains the controlled vocabulary may decide not to not to have "Account" as atomic element in the vocabulary, because it's too general, preferring in this case to have atomic elements that consist of more than one word, such as "Savings Account," "Checking Account," "Current Account," and so on.

Note also that in Figure 3 the example data element and the name of its datatype ("Local_Instrument" and "Instrument_Code" respectively) literally contain the controlled vocabulary terms to which the semantic metadata refers. ISO 11179 and UN/CEFACT Core Components require that the names of data elements that are built in keeping with their specifications literally contain the terms, according to a strict data element naming convention. However, such a naming convention is a non-starter for scenarios where we want to retrospectively apply semantic metadata to pre-existing data elements, which will actually be the most common requirement. Later in this Column I discuss the challenge of finding a place to physically locate the metadata when we wish to apply it to already existing data elements.

More Semantic Metadata – Business Context

There is an additional kind of semantic metadata that is also critically important – metadata that describes the business context within which the data element in question is relevant.[7] Each element of a data structure can be associated with a value, or multiple values, of one or more the following metadata elements:

- *Business process*: The business process(es) with which the element is associated
- *Business process role*: The role(s) directly involved in the business process(es) that are relevant to the element, e.g. shipper
- *Supporting role*: The role(s) indirectly involved in the business process; e.g., a data element in an order response from seller to buyer could be required by a third-party shipper
- *Geopolitical scope*: The economic or geopolitical region for which the element is relevant, such as Brazil or the Euro Zone
- *Industry classification*: The industry or industries for which the element is relevant
- *Product classification*: The product class(es) for which the element is relevant
- *Official constraint*: A legal requirement(s) associated with the element; e.g., an element could be required by Sarbanes-Oxley
- *System capability*: Indicates that the element's purpose is to address the needs of a computing system(s)

When we combine business context metadata with the grammatical semantic metadata described earlier, we create the basis for data integration tools to calculate the divergence between two elements and to trace how the difference in context accounts for the divergence.

---

[6] The 22 Core Data Types are: Amount, Binary Object, Code, Date, Date Time, Duration, Graphic, Identifier, Indicator, Measure, Name, Ordinal, Percent, Picture, Quantity, Rate, Ratio, Sound, Text, Time, Value, Video

[7] These business context metadata concepts are based on an approach defined in the UN/CEFACT Core Components standard.

For example, the two data elements in Figure 4, named Amortized_Instrument_Assessment and FairValue_Instrument_Assessment, share a lot of grammatical semantic metadata in common: They have the same property term ("Assessment"), representation term ("Amount"[8]) and first-level property qualifier ("Instrument"). However, their second-level property qualifiers differ, being "Amortized" in one case and "Fair Value" in the other. Furthermore, the business context metadata tells us that the reason there is a difference between the two elements is that one is aligned with requirements of the US GAPP accounting standards, which allow reporting of amortized asset values in financial statements, and the other data element is aligned with the IFRS international accounting standards, which require reporting the fair value (i.e. the current market value) of the asset.

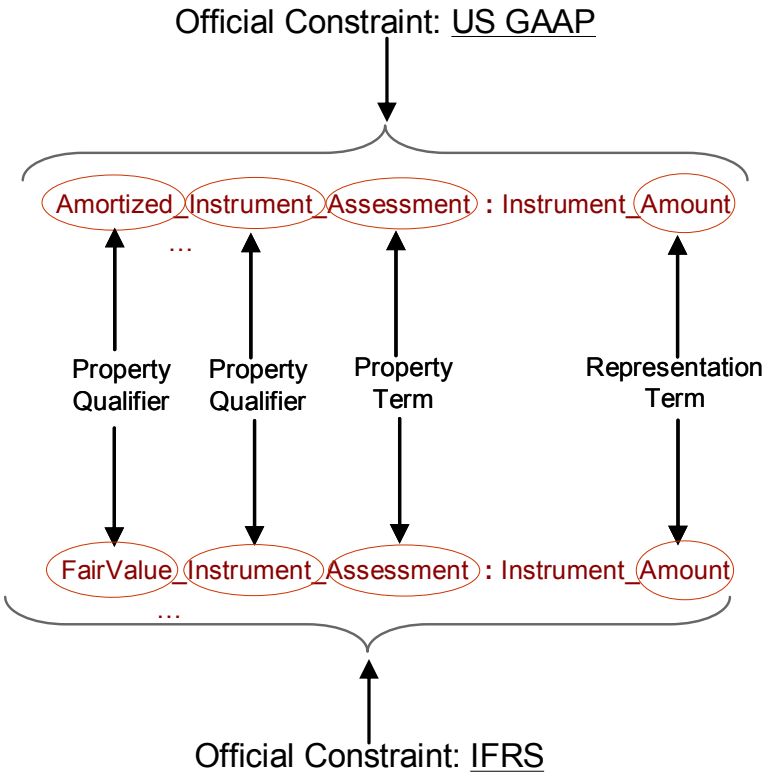

Figure 4: Tracing the Divergence Between Two Data Elements

Putting Semantic Metadata to Work

Let's return to our scenario where the integration analyst is mapping two data formats.

One of the reasons that current-generation data integration tools can produce executable transformations from graphical maps is that they have access to machine-readable *syntactic metadata*, i.e. metadata that defines the syntax of the data structures being integrated. Machine-readable relational schemas, XML schemas, and so on are the syntactic metadata that the tools exploit.

The reason that the tools cannot help the integration analyst decide what maps to what is that there is no machine-readable *semantic* metadata for the tools to exploit. Referring again to

[8] A representation term of "Amount" signifies that, semantically, the element represents a number of monetary elements denominated by a currency, such an amount of Euros. This abstract type contrasts with Instrument_Amount, which is the concrete datatype.

Figure 1 on page 3, the red line between the two "Coupon Value" data elements is meant to signify that the analyst may have erred in mapping the two elements to each other, having been deceived by the fact that the elements have the same name. Today's tools would simply execute that mapping as specified.

By contrast, in Figure 5 below a next-generation integration tool that exploits machine-readable semantic metadata is able to warn the analyst that the mapping of the two elements to each other may be mistaken. The tool points out that, although the two elements have the same name, object class term, object class qualifier, and representation term, the "Coupon" that plays the role of property term for Sales Order 1 and the "Coupon" that plays the role of property term for Sales Order 2 are separate entries in the controlled vocabulary which are related to each other because they have the same English label but have different definitions. The business context metadata explains the source of the difference: One is a coupon for consumer products and one for financial products. The analyst has the final say as to what the mapping should be, so the tool enables the analyst to hyperlink to the definitions of the two separate entries for "Coupon" in the controlled vocabulary so that he or she can make an informed decision.

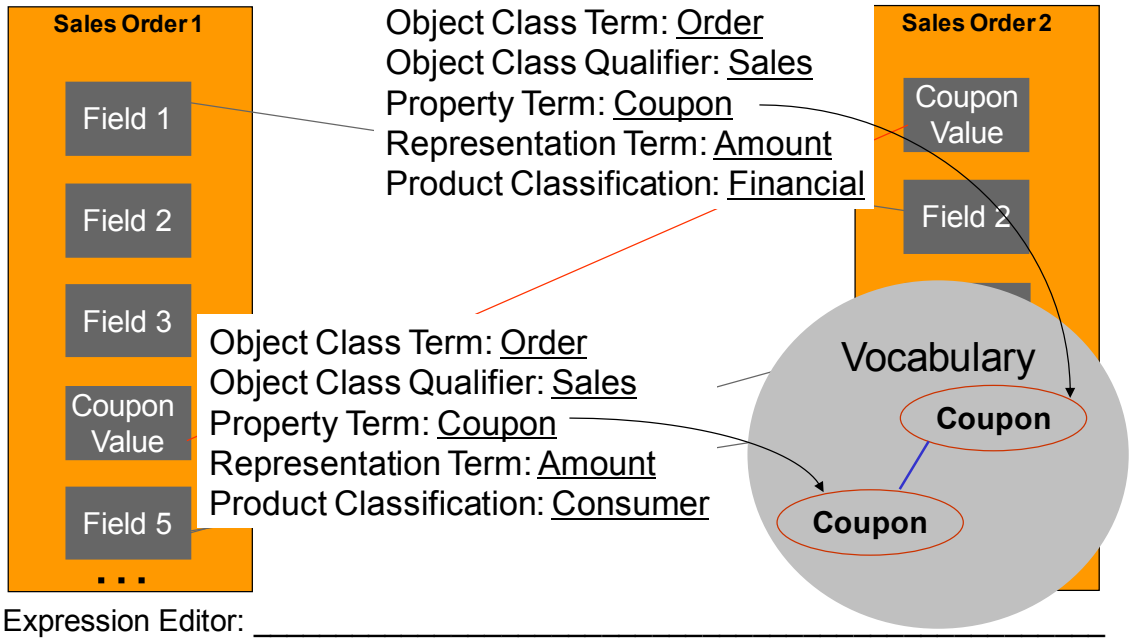

Figure 5: A Next-Generation Integration Tool Exploiting Semantic Metadata

Where to Put the Semantic Metadata – Non-Invasively

As I mentioned earlier, there is an issue as to where to physically put semantic metadata that we wish to apply retrospectively to pre-existing data elements, since the syntactic metadata structures have no slots for this additional metadata. We have to find ways to do this that are non-invasive, meaning that we cannot break applications that depend on the current state of the metadata structures.

The nature of XML makes it generally easier to insert the new metadata in pre-existing XML structures – such as XML schemas and XBRL taxonomies – without breaking dependent applications, as opposed to, for example, relational database schemas which don't have the same flexibility. UML models can also be extended by using UML profiling to define new slots for the metadata, without disturbing applications that read the models. In the case of relational schemas, which are more fragile in this respect, the semantic metadata may have to be located

outside the affected schemas, and the new metadata elements will have to have references to the schema elements to which they apply.

Beyond Manual Markup

The sheer volume of data structures defined in modern corporate IT systems can overwhelm manual efforts to retrospectively mark them up with semantic metadata. Thus, it is necessary to focus on the most important data structures initially. Manually marking up the structures requires that domain experts enter this knowledge via the integration tool's user interface.

Some of the emerging next-generation data integration tools have a nascent ability to read written documentation of data structures and produce the semantic metadata. This more ambitious approach is still somewhat experimental, but the tools will improve this capability over time.

Who is Using Semantic Metadata?

As I've written in my previous articles, this approach to semantic metadata is being used to varying degrees by a number of standards initiatives and major ERP vendors. In the standards arena, projects are underway to address this problem in several industrial sectors (see Figure 6):

- *Manufacturing sector*: An increasing number of standards bodies for the manufacturing sector are collaborating by using a common methodology and a common repository hosted by the Open Applications Group (OAGi). The common methodology is based on UN/CEFACT Core Components.
- *Finance sector:* Under the auspices of ISO's Technical Committee 68, which has jurisdiction over standards for the financial services industry, multiple finance standards bodies are collaborating on a common methodology and repository. The ISO 20022 standard defines the architecture and governance procedures for this alignment project, and the popular ISO 20022 payment messages are just the first content to populate the common repository. New extensions to ISO 20022, which cleared the last ratification hurdle in 2012, allow for semantic markup of data elements.
- *Retail Sector:* The retail sector's standards bodies are mostly grouped within the GS1 umbrella, and the GS1 XML project is aligning with the Core Components methodology.

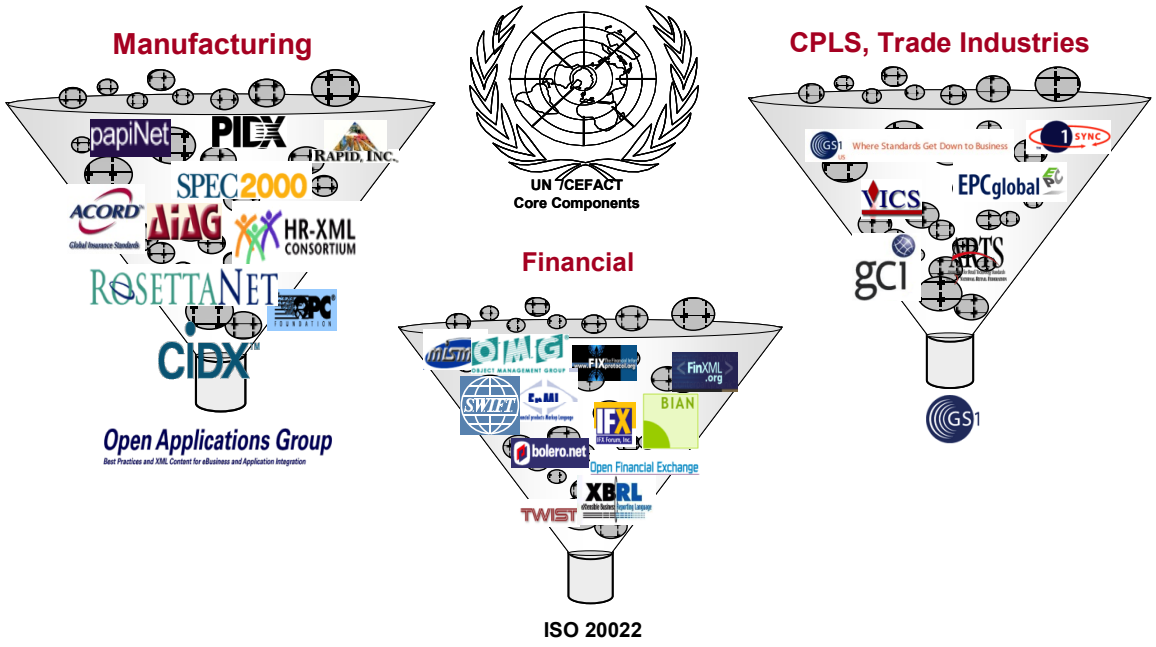

Figure 6: Semantic Metadata in the Standards Arena

Conclusion: Making Ontologies Actionable

We are at the beginning of an era in which we will make measured but steady progress in overcoming barriers to productivity, through the use of semantic ontologies that encode human domain knowledge. Semantic metadata is the key bridge between semantic ontologies and IT data structures, the link that enables a fledgling next-generation of data integration tools to use the ontologies to improve semantic interoperability and break up data integration bottlenecks.

David Frankel has over 30 years of experience in the software industry as a technical strategist, architect, and programmer. He is recognized as a pioneer and international authority on the subject of model-driven systems and semantic data modeling. He has published two books and dozens of trade press articles, and has co-authored a number of industry standards including XBRL, ISO 20022, BIAN, and UML®.