# *Ontology Engineering: An Application Perspective*

Salim K. Semy, Kevin N. Hetherington-Young, Steven E. Frey
The MITRE Corporation
202 Burlington Road
Burlington, MA 01730
{ssemy, khetherington, sfrey}@mitre.org

## *Abstract*

*Enterprise Integration is a key challenge faced by many organizations, including those of the U.S. Government. An important enabler of enterprise integration is data integration. There is promise in applying Semantic Web technologies to enable data integration of disparate data sources based on the semantics (meaning) of the data. As we start to build ontologies within United States (U.S.) government domains for this purpose, we consider how Semantic Web technologies may operate within existing operational infrastructures and how best to build ontologies to facilitate usability. This paper explores the ontology engineering and adoption challenges based on current state of the art in Semantic Web technology. We consider alternative approaches to engineer ontologies, discuss current and emerging standards in this area, look at approaches to integrate data through ontology mapping, and outline a set of skills necessary to develop and apply ontologies. We then propose an architecture to insert Semantic Web technologies into an existing infrastructure to enable data and application interoperability, based on experimentation within the Intelligence Community. The paper concludes with recommendations on a way ahead, and highlights some potential pitfalls for future ontology developers.*

## *1. Introduction*

With the increasing maturity of Semantic Web technologies, we are at an opportunistic time to understand how best to apply these technologies to application areas within United States (U.S.) government domains. A key challenge that these organizations face is Enterprise Integration within and across organizational boundaries. These issues are underscored with the existence of heterogeneous business processes, policies, data sources, and tools. While certain problems, in addition to technical, involve political changes, it is important to consider and evaluate technologies that overcome the technical obstacles.

A key component to Enterprise Integration is integration of the underlying data sources. Semantic Web technologies appear to offer a promising way ahead to facilitate integration. However, there is still an open question of how best to apply these technologies within the U.S. government. Analogous to software engineering, ontologies

follow a development, deployment, maintenance, and replacement lifecycle. What is involved in each phase of this process? What are some best practices and considerations that one has to make before embarking into ontology adoption? Where do ontologies provide short-term gain and how do we inject this technology into existing operational infrastructures?

The focus of this paper is to provide an application perspective on ontology engineering and adoption of Semantic Web technologies for immediate use and return on investment. We start by highlighting considerations one has to make before starting to develop ontologies. This includes background necessary to develop ontologies, standards you should be aware of before starting the development, different approaches to develop and map across ontologies, and tools and frameworks that may be useful in the development process. We then delve into a case study, based on our experimentation, which looks at enabling data integration and tool interoperability within the Intelligence Community. While our discussion is focused on a specific use case, we argue it can be generalized to other application areas and provides a good reference implementation. We conclude with some highlights of lessons learned, best practices, and a promising way ahead.

## *2. Background*

### *2.1 Semantic Web*

The Semantic Web is defined as "an extension of the current Web in which information is given well-defined meaning, better enabling computers and people to work in cooperation" [1]. While the current World Wide Web provides access to an enormous amount of information, it is currently only human-readable. The Semantic Web, incrementally building on top of the current Web, allows expression of the semantics of this data precisely enough so that it is machine interpretable. This facilitates the use of agent-based technology to better mine and filter information needs expressed by information consumers.

Semantic markup of the data is achieved through the use of ontologies. Within the Web community, an ontology is a document or file that contains a set of resources and relations between these resources, defined as a series of triples. Such triples may be represented either as (Resource, Property, Resource), defining a relation between two resources, or (Resource, Property, Literal), assigning a property value to a property associated with a particular resource. Ontologies, with the addition of standards to represent ontologies, such as RDF and OWL, provide a mechanism to represent semantics of data in a uniform way. This facilitates machine to machine (M2M) interaction and data integration.

Furthermore, ontologies also provide a mechanism to allow inferencing on the data, such that an inference engine, in combination with rules, can derive new facts and conclusions implicitly represented in the data.

### *2.2 Use Case: Counter-Terrorism*

There is a compelling case for semantic integration of data from multiple sources to support counter terrorism. A variety of U.S. government organizations within the Intelligence Community (IC), Department of Homeland Defense (DHS), Department of State, and Department of Justice perform functions related to counter terrorism. Many of them are creating, populating and maintaining databases which contain information on known and suspected terrorists, terrorist organizations, networks, travel, communications, finances, activities, etc… In most cases the information is derived from a few primary sources, reported through intelligence or law enforcement channels, and then entered multiple times into databases tailored to serve local purposes or functions. The multitude of terrorism databases that are proliferating do not share common data schemas, standards, or structures, are maintained at differing levels of fidelity, and often contain conflicting information. Furthermore, these databases are often individually interfaced to a diverse array of analysis tools, and applications. There is arguably a tremendous amount of redundancy in the effort to support the many organizations that need access to essentially the same data but for different purposes.

Some initial efforts have been made to improve the availability of counter terrorism information but the are largely organizational in nature, vesting responsibility for consolidating and validating data into a few offices. For example, "The government established the Terrorist Screening Center last year to consolidate 12 existing government databases on suspects into a unified watch list that local, state and federal law enforcement authorities can access." [2] "The creation of the TSC does not provide any new law enforcement or collection powers to any government official; it simply consolidates information that law enforcement, the Intelligence Community, the State Department, and others already possess and makes it accessible for query to those who need it - federal security screeners, State and local law enforcement officers, and others." [3]

The opportunity for application of semantic technologies to address this data aggregation and integration challenge is excellent. Our goal was to allow a diversity of counter terrorism data producers and end users, who have tools that support their needs in a locally optimal manner, to interact with a larger and more comprehensive dataset. Semantic technologies provided for the aggregation, translation and mediation of meaning across a heterogeneous array of databases. As part of our applied research effort, we used a commercial ontology management system to map semantically equivalent concepts contained in a number of counter terrorism databases to a higher level ontology. The resulting ontology and underlying databases was termed a 'Knowledgebase' and provided a method to aggregate available information on terrorism from multiple sources. The individual counter terrorism analysis tools remained unchanged and continued to write to their proprietary databases. We configured the counter terrorism analysis tools to read in data in a format that mirrored their native schemas from the knowledgebase. This architecture allowed us to demonstrate interoperability across an enterprise employing a heterogeneous array of tools and databases. The ontology layer effectively allows individual analysts to continue to use

their tool of choice without modification, and provides the community of analysts enhanced information by spanning multiple databases.

The real issues of security policy, intelligence releasibility, and need to know were not addressed by this research. Questions of how classified data markings and tagging can be used to facilitate sharing across security boundaries are left to researchers in the field of multi-level security. For the purposes of this project we assumed 'system high' level classification allowed for open access to all information contained in the knowledgebase.

## *3. Ontology Engineering Considerations*

In the process of developing ontologies, irrespective of the intended purpose for the ontologies, there are a number of considerations that one has to make. Such considerations range from how best to develop the ontologies to how to deploy and maintain them once in operation. This section outlines some of these factors and choices associated with each.

### *3.1 Ontology Standards*

#### *3.1.1 Ontology Representation Languages*

While the language used to represent the ontology may be determined by the tools used for development, it is important to realize the choices available and emerging standards that may impact the utility and support for the ontologies in the long-term. Most web ontology languages are founded on Resource Description Framework (RDF) [4] and its subsequent extension, Resource Description Framework Schema (RDF/S). RDF is a general purpose language for representing information on the web, built on URI and XML technologies.

More recently, Web Ontology Language (OWL) [5] builds upon and provides greater expressiveness than RDF and has become a standard recommended by the World Wide Web Consortium (W3C). OWL has three increasingly expressive sublanguages, OWL Lite, OWL DL, and OWL Full. At present, small subsets of tools have support for OWL Lite and OWL DL and even fewer support OWL Full. Nonetheless, the tools are emerging rapidly.

#### *3.1.2 Ontology Query Languages*

There are a number of emerging specifications that provide a syntax to query ontologies. However, it appears standardizing on this syntax is lagging behind the ontology representation languages. A number of standards have been proposed, including RDF Query Language (RDQL), DAML Query Language (DQL) and its successor OWL Query Language (OQL). None have been accepted as a W3C Recommendation to query ontologies. Others have applied languages that have been traditionally used for other, but

related, purposes, such as XQUERY used to query XML documents. The choice of query language is often tightly coupled with the Ontology Management System of choice.

### *3.2 Ontology Development Process*

Before embarking on developing ontologies, whether the aim is to provide a semantic model for a particular domain or facilitate data integration, one has to decide on a process. There are a number of alternatives in this regard, including a top-down and bottom-up approach. Each has pros and cons and may be more or less preferable depending on the application of the ontologies.

Top-down ontology development implies an ontology is developed to characterize a domain or community of interest first, based on top level concepts within the domain. This is followed by a process by which this domain ontology is extended and further developed to map to more specific constructs, finally reaching the point where there exist direct mappings to data sets within the domain of interest. Such an approach forces developers and domain experts to think through the characterization of their particular domain and identify key concepts and relationships. Practically, however, reaching agreement across the domain experts is a challenge. Nonetheless, if and/or when this obstacle is overcome such a semantically-rich domain model provides a clear characterization of the domain, which can be subsequently used to map concepts across domains.

In the process of developing ontologies from a top-down perspective, it may be useful for ontology developers to build their domain ontology leveraging and reusing existing ontologies that have been developed by others, such as foundational ontologies [6]. Such an approach allows one to not only reuse existing ontologies, but also forces one to more exhaustively think through the representation of the domain and eventually result in a more complete and concise ontological model. Building on the experiences and work of others, domain ontologists have the flexibility to extend concepts within a foundational ontology to best suite their purpose. While this is still an open issue, the hope is reuse of common foundational ontologies will also aid in the mapping process across domain ontologies.

A key challenge with this approach is extending the domain ontology such that lower level concepts share a strong correspondence with data sources of the domain, while still maintaining model integrity. This emphasizes the need for awareness of the data sources within the domain in the process of developing the higher level domain ontology. Ontology development should involve people with wide breadth as well as depth of domain knowledge.

A top-down approach appears to work well if the goal is to provide a characterization of a particular domain and use this as a mechanism to reach a consensus across domain experts.

An alternative approach for developing ontologies is to use existing data sources as a starting point, developing ontologies specific to the individual data sources first, and then continuing to link and extend these ontologies moving toward a greater level of abstraction. This approach, the bottom-up approach, clearly ensures that the data sources are accurately represented, as they serve as a foundation for the domain ontology. Theoretically, since the information produced and consumed by the domain area serves as a good representation of the domain, such an approach would develop an accurate model of the domain. Consequently, however, inconsistencies and implementation details are made visible in the higher level domain ontology. As a result, the domain ontology may not provide a complete and accurate depiction of the domain and relationships within it, as the ontology is focused on data source specificities.

Once the lowest level, data source specific ontologies have been developed, it may be useful to reuse ontologies developed by others as one starts to develop more abstract models of the domain. Such ontologies, known as utility ontologies, are concepts that are prevalent across domains and can be used as-is or extended to represent these concepts in a particular domain. Examples of such utility ontologies include Time and Space. Use of such ontologies does, however, depend on the linking strategy one applies.

In general, it appears bottom-up ontology development is better suited for data integration experiments, as the result of the ontology development leaves behind an accurate model for the underlying data sources that are being integrated.

### *3.3 Ontology Mapping*

Once ontologies for a domain are developed, regardless of the approach taken to develop the ontologies, linking concepts across these ontologies is an important aspect of semantic integration. This holds when one is mapping within a single domain or across domains. As with development of the ontologies, there are alternative approaches to do ontology mapping. Two approaches discussed in this section include mapping to higher level reference or upper ontology and direct mapping, through functions such as equivalence and subclass, inherent in the ontology representation languages.

Reference ontologies represent abstract concepts, which may be used to relate common concepts across multiple ontologies. Such ontologies are often developed by outside parties and reused and extended by many. As mentioned in the previous section, foundational (also known as upper) and utility ontologies are common examples. Looking at an example where one is integrating two data sources, one data source may contain information about “individuals” while another may contain information about “person of interest”. A reference ontology generalizes these data source specific concepts into a super class, perhaps “person”. Mapping the dataset specific concepts to generic concepts within a reference ontology allows one to relate instance data from disparate data sets. For example, if both individuals and person of interest classes are declared as subclasses to person, a data consumer can now query the data sources at a more abstract level, namely ask for all entities belonging to person, and retrieve instance data across the two data sources. Furthermore, overlapping properties between the lower

and upper concepts that may also be stated as equivalent facilitate fusion of the data into a unified representation.

A second approach to linking ontologies is through direct mapping of concepts across the lower level ontologies. In the example above, instead of mapping an "individual" and "person of interest" to "person" contained within the reference ontology, individual and person of interest may be related directly to one another, perhaps through an equivalence function. In this case, a data consumer querying on individuals will also get relevant instance data from person of interest, as they are stated equivalent. This approach, however, is only applicable in cases where there is a true equivalence between two concepts, i.e. every member in one class must also be a member of the other class. In the case where such a relationship does not exist, other relations, such as subclass or user defined object properties, may be used to relate concepts across ontologies.

While semantic linking through direct mappings may appear natural when mapping within or across domains, there are issues of extensibility with this approach. Looking back at our example above where we've stated equivalence between "individual" and "person of interest", let's assume we have a third data source which contains a concept of "employee". In order to accurately map this with the existing ontology, one has to consider the relationship between "employee" and "individual" and "employee" and "person of interest". An "employee", for example, may be an "individual", so setting equivalence between the two may be ok, but not necessarily a "person of interest". The transitivity property does not necessarily apply in all cases. At the same time, however, stating that both "employee" and "individual" belong to the same class of "person" is not as precise a statement as saying they are equivalent.

### *3.4 Ontology Construction and Deployment Tools*

To facilitate the construction of ontologies, there are a number of existing tools, and more are emerging, that provide a framework to construct and deploy ontologies. These are known as Ontology Management Systems (OMS). OMS tools are comparable to Integrated Development Environments (IDE) often used by developers in software engineering. OMS tools allow ontologists to construct ontologies, most with a Graphical User Interface (GUI). Furthermore, some tools allow one to map ontology concepts to existing, external data sources, such as databases and web services. Upon completion of the ontology, the ontologist can further deploy it to a container for the ontology, such as an inference engine. The inference engine serves to not only store the ontology, but also contains intelligence to further reason on the ontology and infer new facts and relationships. Inference engines are available in varying degrees of functionality and complexity. Finally, once the ontologies have been deployed, a consumer of data, whether contained within the ontology or externalized and mapped to the ontology, can query the inference engine and retrieve data based on semantic queries. Often, OMS tools provide an integrated capability to query the inference engine from within the OMS environment.

### *3.5 Ontologist Skill Set*

Ontology development really requires a marriage between one or more Subject Matter Experts (SME's) of the domain and a technologist. The SME brings the domain expertise to the table, participating in the characterization of the domain or providing insight into the semantics of the data within the data stores. This is essential if one expects to develop an accurate depiction of the domain, including the intended and actual use of the data, as it pertains to the particular domain. The technologist, on the other hand, must have knowledge and understanding of the various considerations outlined earlier in this section. For example, he must understand database technologies along with the database schemas. Schema understanding must include how the databases are used and populated, not just logical and physical data models. How are particular fields populated and how might that deviate from the original database design? How has this use changed over time? What other non-RDBMS data sources are being mapped and how is that data stored and organized? Regarding Ontology development, the technologist needs to not only be familiar with the tools he/she is working with, but also have a good understanding of the vocabulary of the ontology representation language, the intricate considerations one has to make in the development process, and a vision of how ontologies will play within the existing operational infrastructure. To restate the last point, the technologist should have background in the current technical processes that are implemented within the community of interest.

## *4. Data Integration to Support Interoperability within the Intel Community*

An effective knowledge base allows us to capture, manage, and present disparate interconnected data in a unified way. This aids collaboration of distributed users as they now have access to common information sources. We set out to demonstrate the knowledge base as a framework that easily adapts to both new data sources and new consumer applications, facilitating data discovery and integration.

In order to test the usefulness of the knowledge base, we set out to facilitate interoperability between two link analysis tools - *Analyst's Notebook* [7] by i2 and *VisuaLinks* [8] by Visual Analytics. These tools are both in active use within the Law Enforcement and Intelligence Communities and represent a good test case for how Semantic Web technologies could facilitate data, and consequently tool integration within an operational setting. Normally, each tool connects to its own stove-piped database that we would like to expose to the other tool, and after integration, should be able to make use of and share the data in the repository.

While one of these tools does have an Application Programming Interface (API), we did not feel such an approach would be extensible for our purposes. Instead, we chose to integrate the tools through back-end data integration maintaining out-of-the-box tool functionality. Our approach was to construct a common knowledge base that would serve as a collaboration framework for consumers of disparate data.

Since one goal was to build our knowledge base to be usable by the broadest set of tools possible given the state-of-the-practice, we chose to build it around an inference engine that feeds a relational data mart. This data mart represents a relational model of the reference ontology. Existing tools can access the relational data mart while tools capable of generating semantically rich queries can interface directly with the inference engine.

In this section, we provide details of our implementation approach and highlight our lessons learned.

### *4.1 Existing Tools and Data Sources*

As we mentioned earlier, through the course of this experiment, we were primarily working with two tools used by Intelligence Analysts for link analysis, *Analyst's Notebook* combined with *iBase* [7], both by i2, and *VisuaLinks* [8] by Visual Analytics. The goal was to enable data exchange between these tools transparent to the users of either tool. Furthermore, we made a design decision to not build any custom software components that did not allow immediate out-of-the-box use of the tool.

As far as data source connectivity, both *VisuaLinks* and *Analyst's Notebook* are designed to interface with relational databases via ODBC and JDBC and require structured, relational data. They will not integrate directly to our inference engine, which accepts XQUERY and returns XML. While *VisuaLinks* is data source agnostic, *Analyst's Notebook* requires a specific database structure. Therefore, *Analyst's Notebook* was primarily intended for a particular database with particular structure and content. For the purposes of this paper, we'll refer to this database as DB1. *VisuaLinks*, on the other hand, did not require a particular database structure. However, it was a cumbersome task to model the particular structure within *VisuaLinks* to allow the tool to visualize the information from the database. For our experimentation, we chose a particular database that was suitable for our use case scenario. This paper will refer to this database as DB2.

Both databases, DB1 and DB2, contained data to support link analysis. So, each database contains entities such as people, locations, vehicles, and organizations that have certain interesting characteristics and various types of links to each other, such as *memberOf* and *associatedWith.* Across both databases, there are both overlapping and unique entity types. Furthermore, there also exist common instances across these databases, although represented in different ways.

In addition, only *Analyst's Notebook* offers an API from which we could dynamically capture user queries. However, even in that case, the tool does not provide a mechanism to generate semantically rich queries.

### *4.2 Semantic Web Technologies*

For the purposes of our experimentation, we chose to use a suite of tools by Network Inference. *Construct* is a Microsoft Visio plug-in used to construct and test ontologies represented as OWL, i.e. it was the OMS. *Cerebra Server* is a DL-based inference engine that provided a persistent storage of the ontologies in addition to inferencing. The tools supported mapping between ontologies and externalized data sources, such as

relational database management systems. Cerebra Server accepted queries to the ontologies, represented as XQUERY, and returned the results in XML form.

While there were other available tools for implementation, at the time of this experiment tools by Network Inference appeared the most promising for our purposes. For example, it was one of few that supported the recent OWL standard. Since then, other tools with OWL support have emerged, including Jena2 and Protégé. Theoretically, the proposed architecture allows substitution of other implementations of OMS and Inference Engines.

### *4.3 Integration Architecture*

Without the knowledge base, each tool would normally connect to its own set of databases, forming application stovepipes. One common solution is to connect each end-user visualization tool to the same set of databases, but this requires new mappings between end-user concepts and database objects for each application-database combination. Each time a new data source is identified and needs to be merged, a new set of application-specific database mappings needs to be built. In addition, it is more difficult for the user to identify data across databases that can be merged. Alternatively, the applications could connect to a data warehouse. However, this requires a costly warehousing effort and limits the ability to quickly add new data sources.

By building a knowledge base architecture around an inference engine, the end-user tools have a single data source – the reference ontology. As new data sources are identified within the enterprise and mapped to the reference ontology, all of the visualization tools will immediately pick up the new source. With such an architecture, the only mappings required are to first build a database-specific ontology and then map that to the reference ontology.

#### *4.3.1 Use of Ontologies*

Since a key motivation for building this knowledge base was semantic integration of disparate data sources, we needed a mechanism to provide semantic markup to the data sources and semantic linking between the semantic markups. We chose to do this through ontologies.

Our initial approach was to develop the ontologies in a top-down fashion. We started to build an ontology to characterize the domain, initially at a high level irrespective of the data sources. A primary motivation to choose this approach over the bottom-up approach was we did not know the specifics of the data sources (databases), structure or content. Upon identifying specific data sources for our experiment, however, we quickly realized discrepancies in the representation of key concepts and fuzziness in the domain ontology which would have direct implications on the success of the integration. Therefore, we decided to switch our approach to a bottom-up approach, starting with the data source ontologies first.

As we started to develop these ontologies, we found the structure of the data source mapped closely to our ontology representation. While this happens naturally, we found it

later to be quite useful, as the ontology provided a cleaner interface of the data source. At this point, however, the semantics of the data were not yet visible. This indicated that the ontology development should follow a layered approach, as illustrated in Figure 1. Employing a bottom-up approach to this methodology, the bottommost layer of the ontologies serves as a common interface to extract instance data from the respective data sources. As mentioned, the ontologies of this layer quite closely follow the structure and content representation of the individual data sources. This does, however, eliminate the heterogeneity of the data access mechanism to the underlying data sources. The second tier of this ontology stack abstracts the data into more wide-ranging concepts. Classifying lower level concepts into these higher level classes helps to associate concepts across ontologies. While semantic richness at this level is still minimal, it does provide a single point of entry to access multiple data sources. The third, and final, tier of the ontology stack is the contextual stack. It is this layer that provides the greatest level of semantics of the data, allowing one to characterize the domain and represent the usage of the data within the particular domain. This layer, along with the second layer of that stack, is where associations may also be made across domains.

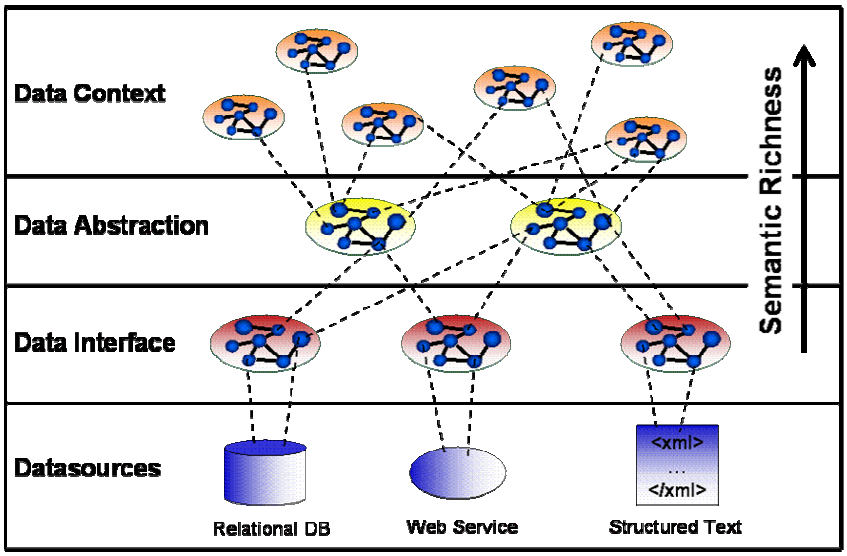


*Figure 1. The Ontology Stack – a layered approach to ontology development*

*4.3.2 Integrating existing user applications*

In our case, the commercial end-user tools are not semantically-enabled and require relational data sources. For this reason, we built a relational data mart representing an RDBMS version of the reference ontology, which serves as a JDBC/ODBC knowledge base interface for the client tools. The data mart is updated periodically by querying the inference engine, at which time the RDBMS-enabled tools have access to the data. Tools capable of generating semantically-rich queries still query the inference engine directly. Figure 2 provides a view of the architecture used in this experiment.

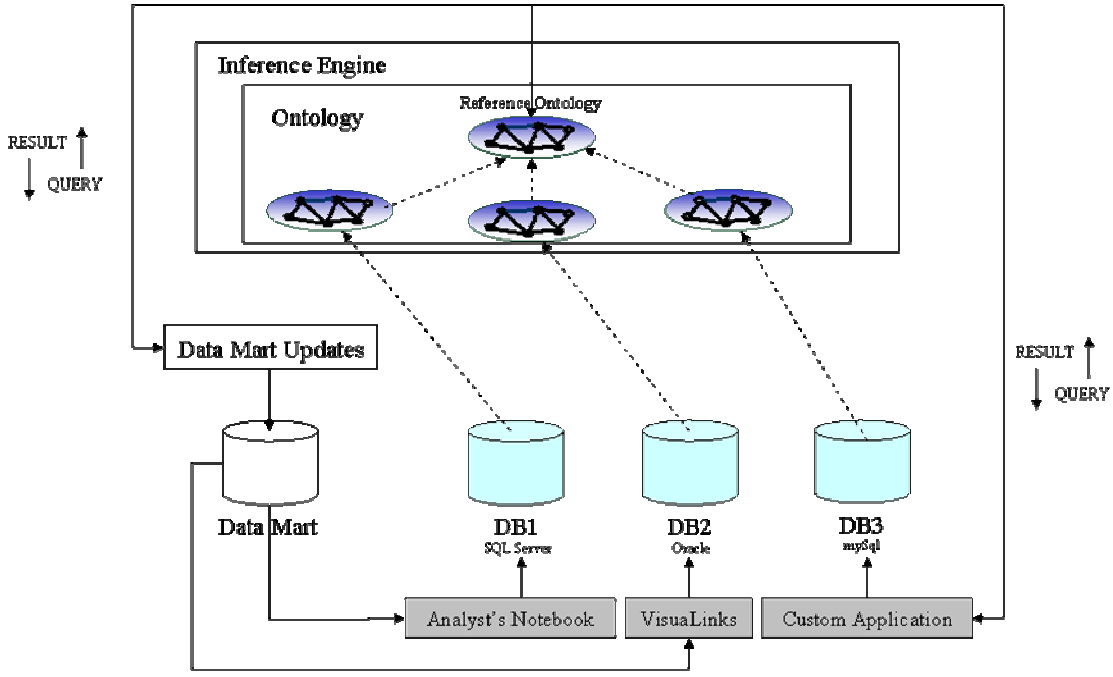


*Figure 2. Architecture supports legacy applications that require relational data sources and applications capable of generating semantically rich queries.*

*4.3.3 Evaluation of our approach*

One cost of this approach is that it requires a duplicate data store. As long as we have client applications that require structured, relational data, we need to build the data mart. This solution is more than a glorified ETL tool, however. Going through the ontology layer and inference engine still gives us the ability to integrate data in semantically interesting ways, depending on how it's modeled.

However, the requirement that at least some of our end-user tools must connect to relational data sources limits how semantically rich the data representation is. Our data mart's tight coupling between the reference ontology and data mart represents the middle tier as described above. In this tier the reference ontology brings together our data sources in a single set of wide-ranging concepts, but fails to reach the third tier. It is this tier that allows us to characterize the domain and represent how the data is used within that domain.

The semantic limitations of using the data mart only affect those client tools that require relational data sources. Tools capable of generating semantically rich queries are integrated directly to the inference engine and can take advantage of the top semantic tier that best characterizes the data's domain and its uses.

The architecture allows each tool to plug into the highest level of abstraction from which it can take advantage, allowing us to build ontologies but still use our legacy front-end applications. These applications have an established end-user base and serve specific business functions. This architectural construct is particularly well suited to enterprise domains where there is likely to be a mix of legacy applications and emerging technologies.

At the lowest tier, the architecture also picks up new data sources seamlessly, independent of the number of client applications needing access to the data. After mapping the database ontology and then mapping that to the reference ontology, all applications then have immediate access to the new data source. This avoids time-consuming work and application-specific point solutions, greatly enhancing data sharing across applications.

## *5. Conclusion*

In this paper, we have tried to capture key challenges and considerations that one has to be aware of when applying Semantic Web technologies, specifically ontologies, to an application area. The choices made become essential when one injects these technologies into an existing operational infrastructure. Through our experimentation, we proposed a mechanism to insert this technology and gain short term value, but at the same time maintain flexibility to facilitate extensibility. In the course of this discussion, we've highlighted best practices and potential obstacles. For example, after assembling a team that includes domain experts and technologists familiar with the relevant applications, database technologies, and ontology tools, we found it best to build our ontologies from the bottom up. We also found much value in building an data mart that mirrors the ontology to support legacy applications. Such an intermediate step becomes necessary in

order to migrate enterprise data to a semantic knowledge base. We believe there is value in this approach. Even with the compromises made, much of the benefits of semantically described data passes through and the architecture supports new and future semantically-enabled applications. Nonetheless, there are limitations, partially due to the maturity of the technology and partially due to the state of current tools and their support for Semantic Web technology. However, over time, both the Semantic Web and the current tools will evolve, bringing them closer to seamless integration. To facilitate this, however, U.S. Government organizations need to apply this technology today and push the COTS (Commercial Off-the-Shelf Software) and GOTS (Government Off-the-Shelf Software) tools toward this vision.

## *6. References*


[1] T. Berners-Lee, T., J. Hendler, and O. Lassila. The Semantic Web. Scientific American, vol. 284, no. 5, May 2001, pp. 34–43.

[2] Strohm, February 26, 2004, www.GovExec.com

[3] Fact Sheet: The Terrorist Screening Center, September 16, 2003, http://www.dhs.gov/dhspublic/display?content=1598

[4] http://www.w3.org/RDF/

[5] http://www.w3.org/2001/sw/WebOnt/

[6] Guarino, N. Foundational Ontologies for the Semantic Web, available at: http://zeus.ics.forth.gr/forth/ics/isl/projects/ontoweb/_notes/FoundationalOntologies.pdf

[7] http://www.i2.co.uk/Products/Analysts_Notebook

[8] http://www.visualanalytics.com/Products/Visualinks.cfm