Towards the Enterprise Semantic Web:
Understanding the Evolution of Integration Technologies in the Context of Recent Developments With the W3C Semantic Web Vision

Prepared for the White House Conference Series on Semantic Technologies
Presented at MITRE Headquarters, September 2004

Jeffrey T. Pollock
Vice President Technology, Network Inference
Email: jeff.pollock@networkinference.com

Abstract

The past thirty years of technology evolution drove a series of technical revolutions that repeatedly called into question old methods and produced marked improvements over previously accepted approaches. This paper contends that rising adoption of the W3C Semantic Web language specifications will produce another paradigm shift for software integration and data management tools. Emerging semantic technology approaches leverage model-driven and loosely-coupled data management schemes that can capture business rules, domain knowledge, process models, and everyday business data in reusable formats. These reusable formats, namely RDF and OWL, will enable a new generation of automated tools that can repurpose organizational knowledge to suit changing needs in highly adaptive ways that EAI, EII, and RDBMS approaches cannot. Resulting software architectures will require less code, fewer human resources, and offer improved automation and change resilience.

An organization marches and evolves as a direct function of its successful use of information. Practically a truism in today's pop-business strategy culture; the company's day-to-day operations and long-term success at achieving strategic goals and profitability are enabled by information systems now more than ever. However, after decades of ad-hoc IT growth, there are systemic barriers to any further advancement of core information capabilities using traditional technologies. The old ways just won't work anymore. There is simply too much code, and too little flexibility in software systems today. But all of that is already changing.

Motivation for Change

Three important forces are driving the need for a major shift in the way organizations manage critical business information: skyrocketing data volume, the human bottleneck, and hard-coded software systems. These systemic problems waste untold trillions in duplicative, labor intensive software programming each year.

First, the continual increase in the sheer volume of digital information. The amount of data created in the past three years exceeds the amount of data created in all of human history prior to 2002. Information production rates already outstrip the technical ability to organize, classify and understand content in any repeatable fashion.

Second, the most persistent roadblock to efficient information handling are humans themselves. The arbitrary and inconsistent ways we create digital data have necessitated a deep level of involvement from people at nearly every step. Digital data is rarely constructed in formats that machines can process on their own, and even when data is highly structured, it rarely contains sufficient metadata for automated processing. Involving humans at each step is a time-consuming process that is perpetually lagging behind business changes. In short, the speed of change barrier is inversely tied to the number of people required to manipulate information systems.

Third, technology that can manipulate information without custom code has been inaccessible to most technologists. In pure business terms it can be said that code is equal to dollars. The more code has to be written, the more requirements have to be generated, leading to more software projects and higher operational costs. In technical terms, each interface, API, document format, and data model must be coded to individually. Business rules, workflow, and data meanings are captured in program code that must be compiled, deployed, and maintained inside complex IT ecosystems.

These systemic barriers need not require a wholesale rejection of past technologies, quite the opposite in fact. The decades of IT investments that have transitioned us from proprietary hierarchical databases and anachronistic procedural software languages to SQL databases and Java-based application servers have provided an implementation space that may be easily built upon for further improvements.

Conversely, the solution space has been driven by classical ideas that code is needed to do anything. Code for rules, code for data, code for logic, code for interfaces, and so on. All this talk of paradigm shifts and disruptive technology applies squarely in the IT solution space, where the system architectures and organizational information changes, not the implementation platforms or legacy systems. Fundamentally, the core approach to managing information, rules, logic, interfaces and processes needs to change – not the legacy software platforms which are already in place today.

Before looking at the many faceted values of an Enterprise Semantic Web approach, it will be important for readers to have context into the way integration technology has evolved and taken shape towards its current form.

Revolution in Integration Technology

Like the steady progression of most scientific thinking, integration technology has followed a path of many turns, ups and downs. Revolution occurs not so much in great leaps and bounds, but in the small advances that continually pile upon one another until previously held belief systems crumble under the weight of better methods. Over the past thirty years we have witnessed this process take place within the IT integration space. Today, this steady march of progress continues.

From the very beginning, point-to-point integration has been the default method of connecting enterprise computing systems. Tying IT systems together can be done reliably with a labor-intensive approach of custom-code on a one-off basis. Simply writing new C code, or stored procedures, or Java/XML translators can overcome the basic hurdles to sharing information among different systems.

By the mid-1960's financial institutions faced significant pressure to cross organizational boundaries and connect IT systems for sharing financial data. Electronic Data Interchange (EDI) was born to solve this challenge.

Status quo ensued for nearly twenty years. But as the increasing interconnectedness of institutions, agencies, and business units became a reality – instead of just a novel idea – better tools were needed. Message-Oriented Middleware (MOM) was developed in the late 1980's as a way to guarantee reliability in diverse IT ecosystems. Gradually, throughout the early 1990's MOM transitioned from a raw technology to a sophisticated layering of tools that could manage partner interfaces, do data transformations, handle security access rights, and coordinate message delivery. Enterprise Application Integration (EAI) provided this user friendly packaging for enterprise middleware.

EAI technologies have transitioned over time as well. First, the addition of firewall ready queuing interfaces spawned a middleware package geared for use outside controlled systems – Business to Business (B2B) middleware. Then, with the addition of even more sophisticated workflow technologies for transactions, a new process-centric middleware became prevalent – Business Process Management (BPM).

Today, with the transition from EAI to B2B, to BPM still taking place, business are looking beyond the architectures of message delivery and interface management. More than ever, the importance of shared data; as the primary purpose of middleware in the first place – is paramount.

A new class of technology emerged in 2002 called Enterprise Information Integration (EII). This technology is focused on the monumental challenge of fusing disparate information from culturally divergent sources. The technology uses a combination of novel federated query algorithms, metadata management, and single view schemas to enable business people to quickly get visibility into disparate sources.

EII vendor technology comes in three types of configurations: relational approaches, XML based models, and object-oriented systems. However, the basic architecture pattern remains the same. Start with a large number of disconnected systems, model their interfaces by annotating metadata about what their data means, then create different views of the merged systems based off of the metadata annotations. Finally, EII platforms can reach out an connect to various sources, issues dynamic queries with complex joins, and pull back real instance data that complies to the unified views inside the hub.

It should be unsurprising to everyone that business people care more about the unity and meaning of their data than the technical features of their protocols and message delivery systems. Intuitively, we can understand how better information visibility can enable organizations to model financial scenarios, organize supply chains, coordinate logistics, and generally make

better decisions. It should also be unsurprising that organizations want to avoid being locked into proprietary solutions that constantly require payments to a sole-source provider.

The Drive to Standardize

Harkening back to the EDI era, institutions realized the value of third-party community standards that facilitated open technology solutions, leveled the playing field, and reduced exorbitant maintenance costs.

More recently, the development of Web Services standards, coordinated by the global standards body known as the World Wide Web Consortium (W3C), sought to make EAI a community-owned specification – thereby eliminating a hodge-podge of incompatible middleware approaches.

Web Services combine protocol message services with interface specifications to enable a set of secure, scaleable, and open integration technologies. The promise of Web Services was to enable an era of easy integration with standard architectures, design contracts, and protocol services. Web Services have spawned a new kind of architecture for connecting IT systems. Service-Oriented Architectures, Grid Services, and Composite Application Services are all indicative of this new style of integration – driven by standards and the power of coarse-grained application services.

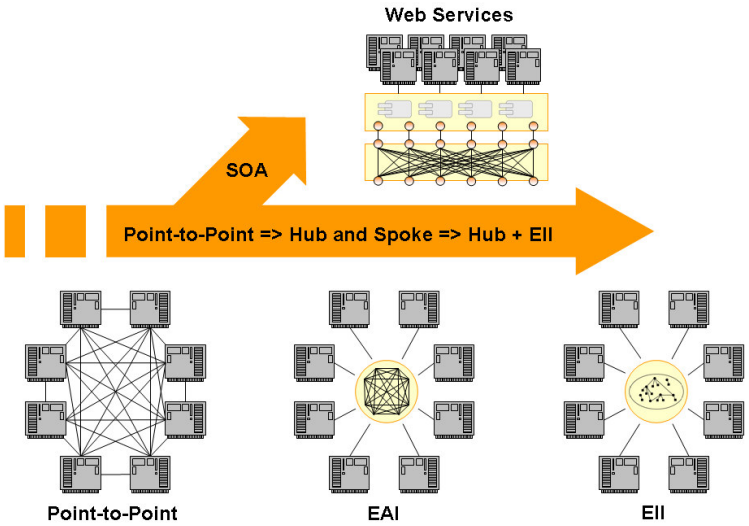


Figure 1 – Migration from P2P, EAI, EII, to standards-based SOA

This revolution is a true breakthrough on one level, and an abysmal failure on another. While there is no doubt about the superiority of service-driven integration, it represents the triumph of coarse-grained interface contracts over fine-grained object (or document, or database) associations memorialized by CORBA style object brokers. However, there are serious problems with any contention that standardized Web Services are a complete solution.

Web Services, Service Grids, Service-Oriented Architectures, or Composite Application Services, were never intended to solve the information level challenges of integration. From the beginning, standards bodies saw the services approach as a way to solve interface, protocol, and delivery issues – not data format, data meaning, or data transformation issues. Thus, even today, with robust service architectures in place, organizations face the very same problems they did thirty years ago – incompatible data.

True, the problem can indeed be solved with proprietary technology like EII, or with brute force point-to-point programming. However, the glaring limitations of these approaches persist – high costs, inflexibility, and isolated islands of information.

Shared Vocabularies, Common Data Models

One way that business and government has tried to combat this problem is in the development of community vocabularies. The popular thinking goes like this: get all the people who care about a shared language in one room, let them drive consensus towards a shared vocabulary and data meaning, then roll the shared vocabulary out to organizations involved in the community. In practice, this seemingly simple process rarely – if ever – works.

Shared vocabularies are perpetually hindered by systemic forces that ensure their failure. For starters, the core conundrum is that for a shared vocabulary to be effective it must be used by lots of organizations – but the more organizations participate in creating a shared vocabulary, the less useful it is because (a) it is overly homogenized and (b) it takes so long to create and change. Evidence of this reality can be seen in the day-to-day usage of large EDI vocabularies, where constituent organizations regularly use "optional" data fields for their most important data because the core specification does not account for the nuance of their proprietary business data.

Further, this approach of standard vocabularies typically leads to a highly centralized, top-down specification that is often policed in a highly rigid manner. Ultimately, the vocabulary becomes extremely difficult to change, nearly impossible to evolve in realtime, and users will often find backdoor ways to get around these problems that cause further interoperability problems.

In the end, nearly all large shared vocabularies are implemented with point-to-point specializations which are non-interoperable with the "standard." Thus, shared vocabularies usually succeed in situations where (a) there are a small number of participants, (b) the domain is small and specialized, and (c) the business processes don't change very frequently. Unfortunately, the majority of global business processes don't match these characteristics.

These problems – with service architectures, proprietary integration tools, and shared vocabularies – are well known in the IT world, despite the fact that the marketing machines of global technology companies continually gloss over them.

Finding a new approach to assist service architectures, eliminate costly and labor-intensive methods for information fusion, and overcome the limitations of shared vocabularies has been top priority for those in the W3C since 2002. Tim Berners-Lee, of Internet fame, has directed the W3C on the Semantic Web – a global effort to enable machines to work with information as seamlessly as people do. The W3C seeks to standardize a new data architecture for all the data on the web. The vision is grand, and ultimately seeks no less than making all the data on the Internet appear to users as a giant database of query-able, categorized, and meaningfully linked information.

While the grand vision may still be years away, with many significant hurdles yet to overcome, the Semantic Web technologies have matured enough to provide breakthrough capabilities for organizations that need large-scale data solutions – with the assurance of a globally backed standard information architecture to support it.

We've dubbed this evolutionary architecture as the Enterprise Semantic Web.

Enterprise Semantic Web

The Enterprise Semantic Web (ESW) is a well-scoped implementation of Semantic Web technology to a limited problem domain. It is more tightly constrained to a domain with well known data provenance, fewer competing contexts, and dedicated modeling resources. Thus, while the big vision of the Semantic Web proper may still be years away, the benefits of an ESW may be realized today. The benefits of an ESW are multi-faceted, but they revolve around the standards-based nature of the approach, the notion of information primacy over physical implementation details, and the agility and adaptability of resulting information architectures.

The business drivers for this new information framework should be clear – low-costs, flexibility, and non-proprietary information visibility. But the technology value is usually less obvious to casual observers of Semantic Web activity.

Precisely speaking, the ESW approach provides a way to accomplish interoperability – not just integration. Typically people use the terms interchangeably, but they mean different things. To interoperate, the parts retain their autonomy, their individual context. With integration, and this is apt, they parts have to become part of the common whole. Fundamentally, this is the difference between brittle codes interfaces and loosely-coupled interfaces for data and connections.

The ESW leverages core W3C specifications to enable any legacy data to easily be converted to flexible schemas, and then create views into that data based on top-down business needs. This is an oversimplified way of saying that the W3C Semantic Web standards will enable data of all formats to be distilled to its core meanings, then reassembled in a variety of ways to meet the needs of specific business applications, middleware, or business processes. In short, the ESW is a new data architecture that can repurpose information on demand, in different ways, and in a highly automated manner.

Resource Description Framework (RDF)

RDF is a W3C standard recommendation approved in February 2004. It provides an easy path to baseline an instance data set to a flexible schema, while preserving the basic semantics required to reassemble the many parts in different "views" using OWL. Originally the specification was designed to represent metadata about parts of web pages for machine processing. More recently, RDF has found broad adoption with news syndication services and people looking for a lightweight method of capturing semantics for just about any kind of data.

The basic data structure of RDF is a "triple." A triple consists of a subject a predicate and an object. Simplistically, the subject uses a predicate to say something about an object. For example, the statement, "John has a friend named Paco," can be represented as a triple where John is the subject, hasFriend is the predicate, and Paco is the object. This simple core structure can be used in a multitude of ways to capture the meaning of most kinds of data.

RDF has been envisioned as an XML compliant syntax, inheriting grammar and important language constructs such as XML namespaces. With RDF data, schemas need not be specified in advance because the schema follows the triples pattern regardless of specialized content. Thus, an RDF system need not require schema changes for dealing with foreign vocabularies from disparate systems.

The use of RDF as a place to store and manipulate data from disparate systems is of crucial importance to the Enterprise Semantic Web vision because data of all structures (relational, taxonomic, object, etc.) can be converted to RDF with minimal loss of meaning. Metadata from source systems can also be expressed as triples right alongside business data enabling an RDF data store to treat data and metadata equivalently. The use of RDF as a way to capture and express the values, instances and metadata of heterogeneous applications becomes core to the ESW vision when it is used in conjunction with another Semantic Web technology called OWL.

Web Ontology Language (OWL)

OWL is a born-and-bred ontology language for specifying domain knowledge, taxonomy, objects, classes, business rules and business logic in a model-based syntax. Layered over RDF, or other data types for that matter, OWL can provide loosely-coupled views of that data completely independent of the native schema. It does so through the use of built in language axioms, or fact statements, about how concepts, instances, and relationships are connected. In this way, OWL ontology fragments can be combined to create a "virtual" schema adapted to particular application or user needs.

Adapting different views of data is a very powerful capability because it enables business people to eliminate the process of building common data models. Traditionally, large IT projects would have to get different political stakeholders in a room to define business terms and vocabularies that everyone agreed on. Now, with OWL, individual ontologies that are accurate within specific contexts can be semantically joined (using axioms) to resolve the naming and labeling semantic conflicts that usually cause community efforts to stall out. Thus, business users can accomplish interoperability by effecting a common data model without the need to laboriously define and maintain it. Further, responsibilities for enterprise data management can be delegated to the key business units that actually define the enterprise's use of data, rather than a costly and error-prone centralized management function.

The most important technical aspect of OWL is that it has a formal semantic model that is grounded upon logically sound mathematics. Translated to business terms, this means that OWL based inferencing, query, and mediation tools will eventually provide the same kind of consistency that we expect from relational databases today. Other ontology approaches lack a formal semantic model, which means that different vendors implement semantics in different ways – ultimately causing a fragmented architecture that is only incrementally better than problems we faced decades ago.

The most important business aspect of OWL is that is allows machines to automate much of the low-level data handling your IT staff would typically have to code for. This concept of machine-actionable semantics is what will enable higher levels of reusability and reconfiguration of business information – automatically.

Reasoning and Inference Services

Reasoners and inference engines are tools that can deduce more about a set of models or data than querying alone. Although there is no universally accepted distinction between a reasoner and an inference engine, most people agree that an inference engine is a kind of reasoner. Reasoners can adaptively reclassify knowledge, rules, and logic based upon external stimulus. From a physical software perspective, reasoners are simple services that can be called at runtime.

Business applications, web portals, middleware and databases can use reasoners to query and (re)classify data while taking advantage of implicit semantics that are deduced from explicit relationships defined in RDF or OWL. This capability enables enterprise software to infer new facts or adaptively change how data is classified – without having to write new code.

Because reasoners work with OWL and RDF natively, they allow data, metadata, business rules, and application logic to be treated equally during a single transaction. This ability to blur the distinction between different uses of data allows reasoners to use all data for answering deductive queries and inferring new facts about a given data set. This power is especially important when considering how disparate ecosystems of ontologies and complex data can be merged dynamically to draw associations between previously disconnected data sets. Thus, the manual and expensive tasks of schema mappings and schema change management can be automated at much higher degrees.

Mediation Services

The mediation service is a more loosely defined concept than inference or registry services. Generally speaking, a mediation engine will act as a service façade that masks the complexity of API calls to various system services. At their core, mediation engines are tools that can dynamically output data in different formats and syntax that comply with disparate schema types. They can enable automated data transformation without code. They can mediate queries to heterogeneous sources. They can also serve as a component backplane for JMX and SOAP managed utilities.

Mediation services a useful for managing transformations when native data needs to be handled prior to exporting to RDF or OWL formats. For instance, exchange services in a Service-Oriented Architecture may require transformations from EDI and relational structures into XML prior to posting data to a Web Service. A mediation engine would help automate these sorts of transformations while preserving data semantics.

A mediation service could also broker queries to a variety of underlying sources. For example, if a user issued a single query that required data retrieval from different Web Service interfaces and also need to make use of various hybrid-reasoning capabilities, a mediation service would broker a complicated multi-stage query execution plan.

Generally speaking, the mediation service would act as a distributed service handling requests for federated query, transform any-to-XML, get status, and examine various nodes within a diverse system of systems.

Registry and Repository Services

Semantic content registries provide a federated registry for the semantics of schemas, ontologies, and applications – they will become the DNS for application semantics. Networked together registries will provide the content management capabilities for data vocabularies, ontologies, canned queries, and other schema metadata. System level needs like versioning, change management, and artifact storage are all features built into most registry engines – such as UDDI and ebXML registries.

Registries are also services that can be used directly by applications and middleware. They are conceived of as linked services that can replicate and share information inside a network of clustered repositories. In this way, a given software application can point to a local registry for content and have access to a wide array of metadata, ontologies, and stored queries that have been duplicated from other linked registries.

Newer registry frameworks being explored by both UDDI and ebXML standards groups are looking to adopt formal semantic data models expressed by OWL. This will eventually enable a rich, semantic query interface to registry contents that could finally deliver on the promise of "dynamic content discovery."

Core ESW Architecture

The first step in assembling these W3C specifications and corresponding tools into a cohesive enterprise architecture is to come to terms with the fact that enterprise data is simply too fragmented to be useful. In other words, enterprise information comes in too many flavors to be effectively managed. RDF provides a solution. RDF can be used as a baseline data format, effectively a normalized lowest common denominator of data within the Enterprise Semantic Web.

As previously discussed, the features of RDF are well suited for its use as a baseline data format. It can represent data from any source (XML, relational, hierarchical, unstructured, or semi-structured) with minimal effort. Simple automated tools can be built to parse legacy data and create RDF triples that capture the original metadata that is present in the source system. The resulting RDF data set has a very simple structure, contains all the data metadata from source systems, and can be federated within distributed RDF databases.

Alone, RDF is a simple way to make data available in a common format – far simpler than XML or relational systems for example. However, it is not very good at organizing data into structures that are self-describing. In other words, RDF is not a semantically rich language that can provide improved end-user understanding of new information. For example, an automatically created RDF database would have to be intimately understood by a data modeler in order to build queries

that extracted targeted information for business use. In this regard it is not much better than older data warehouse schemes that used abstract data models to store arbitrary operational data.

The power of RDF as a base language becomes apparent when the OWL specification is layered on top of it. OWL provides a schema, with rich built-in semantics, to organize and classify the free-form RDF triples into cohesive views that are relevant to business users and applications.

In this way, OWL ontologies can be created in a top-down, business driven, manner to capture and define the way in which data should be understood by machines. Simultaneously, RDF data can be generated in a bottom-up manner with automated tools to extract pertinent data and metadata from an infinite variety of legacy IT systems.

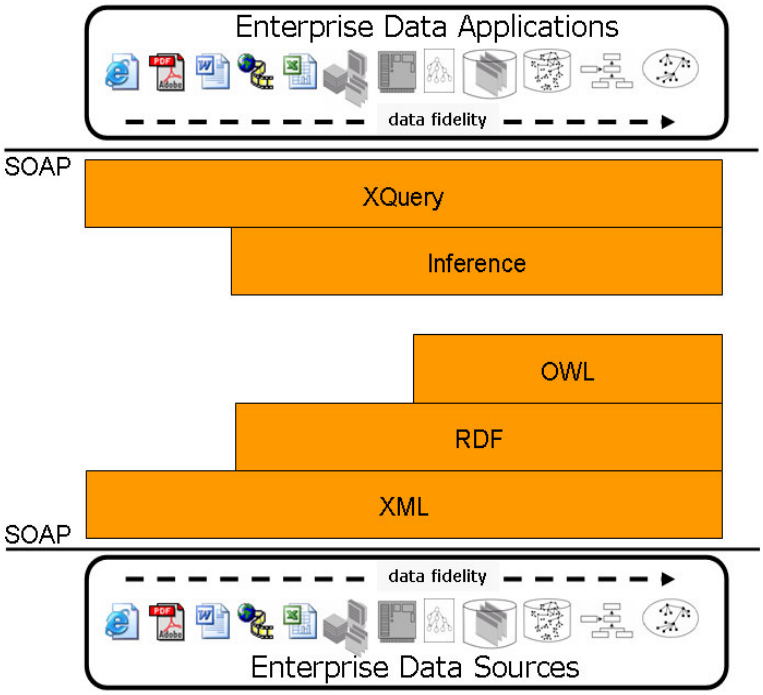


Figure x – ESW Logical Architecture

OWL is linked to RDF via the core W3C paradigm of Universal Resource Identifiers (URIs). Technically speaking, each OWL instance (also referred to as an individual) can be associated with an RDF data element, which is one node of a triple. By connecting RDF with OWL, powerful automated tools can be used to search, classify, repurpose, and merge data sets in realtime, without prior coding.

Using SOAP as protocol layers above and below, a Service-Oriented Architecture (SOA) becomes the de-facto transport and service interface layers around the ESW. Using SOAP in this way allows for rich application connectivity to desktop applications, web portals, operational legacy systems, data sources, and external repositories.

The Service-Oriented Architecture may be straight-forward in nature, but it does not represent the bulk of the work that goes into an ESW.

The bulk of the work in this scenario, which is non-trivial but manageable, is involved in the process of connecting OWL ontologies to underlying RDF data sets. This is the point where top-down and bottom-up synthesis has to occur. Fortunately, tools to help in this process already exist; and are getting better every day.

With the RDF and OWL information architecture in place, tools can be layered in to automate access and improve visibility. The use of inference engines to reclassify data in realtime results in a "virtual schema" for enterprise data. This means that physical data can be dynamically associated with new schema configurations without having to dump and rebuild databases or redefine XML schemas. A mediation service can be layered on top of OWL and RDF to broker queries to each layer, perform transformations between languages, and manage application services for hybrid reasoning engines, federated RDF databases, and query processing utilities.

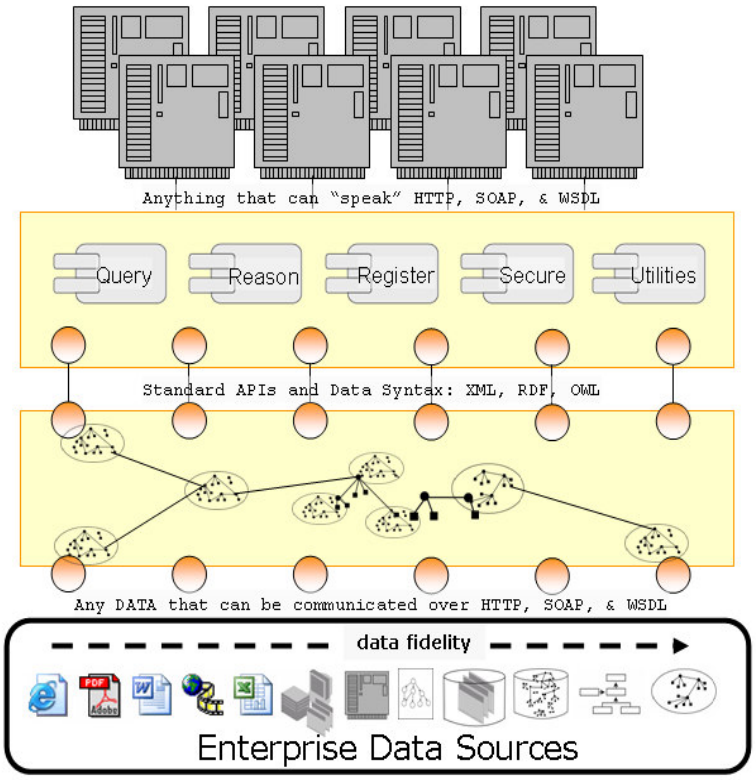


Figure x – ESW Component Layers

This ESW architecture would allow a top-down and bottom-up process to produce actionable information in an automated service-driven IT environment. But this is only a start. By itself, the ESW architecture is of no use to anyone – save a few curious IT geeks. The whole point of this kind of information architecture and tools is to allow business applications to do more with less, adapt to new situations, and drive down maintenance costs for complex systems of systems.

Adaptive IT Solutions

The term "adaptive" is used to denote a capability for machines, without human intervention, to identify and respond to changing conditions. Specifically, for IT systems to sense changes to models (initiated by machines or by humans) and then reclassify data, business rules, and business processes based on the impact of those changes. The Enterprise Semantic Web enables this capability, and this capability can be used in different ways by different kinds of IT systems throughout the architecture.

Adaptive Web Portals

Web-based systems are business applications that are architected around a web paradigm. They will have browser-based interfaces and back-end architectures supportive of either J2EE or .NET frameworks. Web applications can use the ESW to drive screen display, business rules, and

content management of information resources. The ESW is deployed as a service for web applications to query information and business rule models.

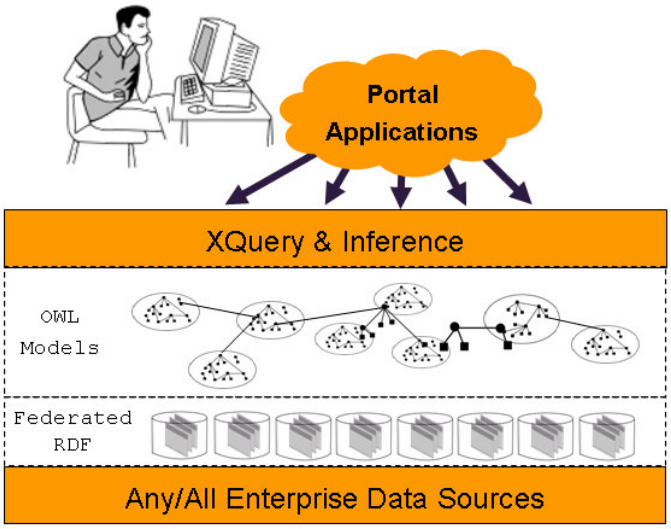


Figure x – Web applications using distributed knowledge

The primary advantage is that web applications can derive their data and rules from a dynamic model-driven environment – rather than coded algorithms, EJB and .NET components. In practice, it will allow business people the ability to make changes to business processes from tools, that in-turn change OWL models, which in-turn reconfigure the way data is combined to specify business rules and enterprise data structures.

Adaptive ERP Applications

Business applications are the big ERP packages that dominate the enterprise landscape within the Global 2000. From financials, to supply-chain, to product lifecycle management and human resources, most large organizations rely on complicated software programs to run their business. Today, these large ERP packages are usually monolithic systems that require significant vendor alliances to customize, configure, and hard-wire changes to support a given organization. Tomorrow, with the ESW in place, organizational knowledge will not be locked into proprietary vendor software, it will live outside the vendor supplied software in reusable models that can be used everywhere corporate data is needed.

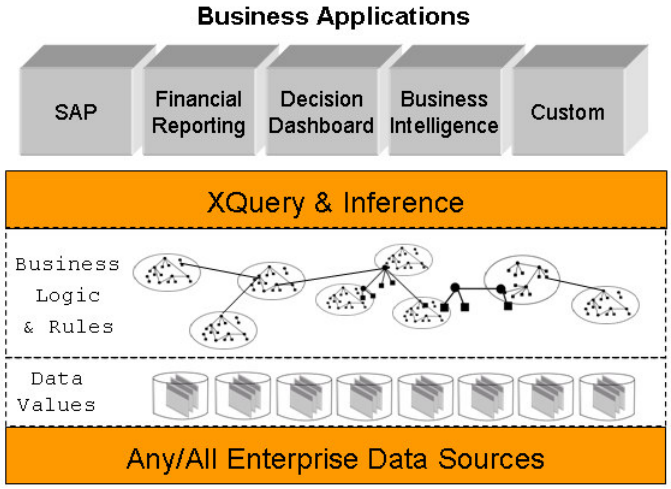


Figure x – Business applications decoupled from proprietary data

The notion of “Adaptive ERP” is the idea that ERP applications should change to meet your business needs, not the other way around. The only way that organizations can ensure that ERP packages will not needlessly tie them into an outdated operational paradigm is to decouple business rules and system data from the software that processes it. The Enterprise Semantic Web enables this vision. By decoupling core business knowledge in models, not in code, it can be repurposed by the many different kinds of ERP systems that might require it. In the future, large ERP vendors will not be able to hold business hostage to perpetual upgrade paths and built-in “best-practice” data models.

Adaptive Middleware

Middleware is the faceless infrastructure that connects pure line-of-business applications into a cohesive ecosystem. Middleware vendors have typically pursued one of two architectural approaches, the enterprise service bus or the hub-and-spoke model. More recently, the shift towards standards-based middleware has driven the industry towards Web Services. In all cases, the crux of middleware issues revolves around data management. Although most middleware vendors and Web Services standards initiatives have solved the connectivity rats-nest that plagued large organizations in the last several decades, the bulk of time and costs for managing middleware today is in the authoring of code for data transformations, vocabulary management, and data interoperability.

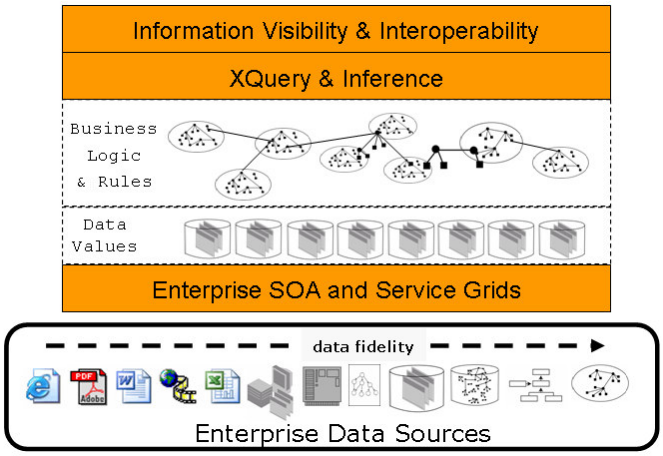


Figure x – Adaptive information, layered over SOA

The Enterprise Semantic Web offers a service-based approach for managing vocabularies, metadata, and data structures outside of vendor-specific middleware tools. Fundamentally different that previous approaches; the ESW relies on a model-driven approach to linking information, describing business rules, and capturing metadata. The reason model-driven works so well for this purpose is that it serves to loosely-couple knowledge from systems. It can decrease the brittleness of interfaces and enable machines to reconfigure services on their own. This is in stark contrast to the layers of code that typically couple interfaces and data packets into inflexible systems that tend to break whenever the business changes.

The bottom line is that traditional integration methods don’t solve the core information engineering problems causing broad systemic pain. An ESW approach alters the old paradigm by enabling users to effect a common data model without the problematic human costs and to dramatically reduce the amount of code required to deploy meaningful integrations on large sets of data. These benefits deeply address major problem areas with the old way of doing things while building upon earlier work.

Adaptive Data Architectures

Fundamentally, the core value of the ESW is in the data layer. But a data layer contains far more than plain data. Anything that can be embodied within software is rightfully considered data. Most people speak towards workflow, metadata, data, business rules, and business process as if they were distinct things. It is true that they exist for widely different purposes, but fundamentally you computer sees them as the same thing: data. By the time that business rules or metadata or process flows make their way into software, they are discrete things that can be processed and modeled the way any data can. In fact, the brilliance of the OWL and RDF technology that underlies the ESW is that they treat it all as data.

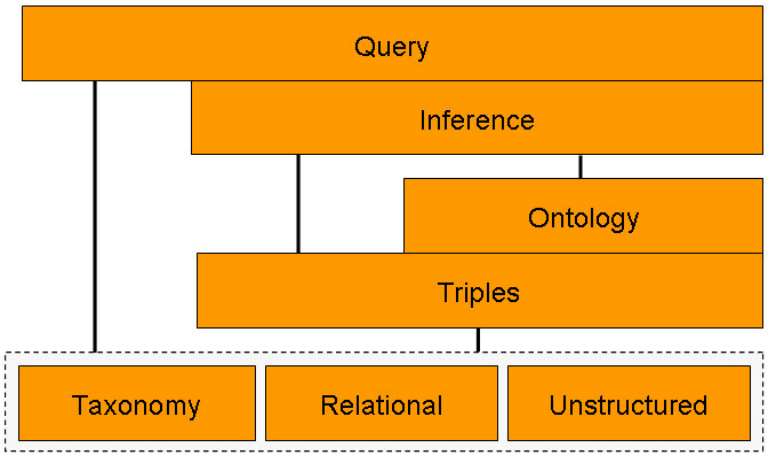


Figure x – Layered data formats of variable fidelity

Business rules can be modeled in OWL. Metadata can be captured in RDF. Workflow and processes can be described in ontologies. Fundamentally, all aspects of the business can be represented within the Enterprise Semantic Web data structures. This is vitally important because it enables common tools (inference, mediation, registry) to work with many aspects of business knowledge at once – instead of pushing things to different software layers with different management paradigms and tools for each part. The ESW successfully confronts IT complexity by isolating the complexities into a single, manageable layer – RDF/OWL.

Adaptive Organizations

The notion of an adaptive organization, sometimes referred to in the industry as an adaptive enterprise, is actually much broader than software systems, or IT in general. The adaptive enterprise is a commitment to a broad range of flexible organizational structures, human resource practices that encourage innovation, business processes designed for agility, as well as corporate information systems that are change resilient and dynamic. The following scenarios, drawn from actual projects within Global 2000 companies and government agencies, should serve to highlight some of the advantages inherent to the Enterprise Semantic Web architectural vision.

Financials

A Fortune 500 electronics manufacturer had trouble reporting quarterly results to Wall Street. The trouble was that their products were always changing and converging, necessitating different algorithms for determining financial results. Like most businesses in the electronics domain, they are faced with a perpetual blurring of lines about what their products really are. Consider the home audio domain, today we can find devices that are DVD players, DVD recorders, VHS players, digital video recorders (DVR), and TIVO services all in one box. How would a large company with tens of thousands of products, discrete and converged, report the sales of the converged devices? Wall Street does not change their predefined market segments quickly

enough to account for new electronics devices, so a labor-intensive accounting process usually has to occur.

The old way of doing things involved a complicated series of web applications written in J2EE that required quarterly code updates, testing, and redeployment. Even with this application in place, the data was usually exported to spreadsheets for further analysis by dozens of analysts. Overall, the process took about six weeks with more than 20 people to generate quarterly statements. Because the process was so human-intensive, reporting errors did eventually find their way in quarterly statements to Wall Street.

The new way of doing things is to connect sales databases, operational databases, and financial systems to an RDF data store. Simple conversions take raw ERP data from a variety of places and reduce it to collections of triples. Then, the RDF data is connected to master ontologies written in OWL. The OWL ontologies represent three fundamental concept spaces: (1) product line hierarchies, (2) financial market segments, and (3) business rules for how products are classified. The resulting solution allows business users to "tweak" business rules and/or product line hierarchies and watch how the actual business data sorts out. At runtime, inference engines are used to query the data and (re)classify how sales results are bucketed into various market segments.

The resulting solution has taken the overall process from six weeks to one day. Errors have been reduced because humans are no longer required to "fat-finger" data into spreadsheets. The value of automated classification and total visibility truly becomes apparent when financial analysts have the ability to test different reporting scenarios and instantly see impacts to the bottom line.

Manufacturing

A top global automotive manufacturer needed to find better ways to rapidly prototype new cars while taking advantage of existing parts inventories. The challenge for car builders today is to be able to innovate faster and come to market with innovations that are cost-effective to build. A significant IT challenge is to ensure, early enough in the design phases, that the parts required for a vehicle can actually fit well with the other parts that are already part of the specification.

Many Product Lifecycle Management (PLM) companies offer capabilities that largely meet this goal. But the rigidity of many of the PLM underlying data models prevents a dynamic environment that would allow engineers the ability to efficiently create ad-hoc designs with existing part specifications.

An early Enterprise Semantic Web architecture was used as a way to enable the dynamic and adaptive requirements of this engineering process. Reasoning tools were used to determine the suitability of various car configurations by using RDF and OWL as core data models to capture the engineering design specifications of various automobile parts and components – independent from proprietary PLM data models.

The value of this use of ESW was to overcome the inherent complexity of engineering specifications and configuration options outside of costly hard-coded ERP/MRP/PLM environments. In fact, most complex configuration-type software challenges can be solved far more efficiently with the ESW architecture than anything else available today. For example, network management configuration, biotech/pharmaceutical drug interactions, and decision support systems with many variables are areas where the realtime adaptability of an ESW approach has proven worthwhile.

Information Supply Chain

A major NATO military force chose to prototype an Enterprise Semantic Web architecture as part of a major initiative under the umbrella of the Network Centric Warfare doctrine. Net-Centric Warfare means that military operations, in battle and for supply-chain logistics, are deeply

connected in ways that improve force projection, flexibility, and coordinated cross-military operations.

The focus of this ESW prototype was on battlespace awareness, also referred to as a common operating picture, that can dynamically show commanders exactly what is happening in battle. The ESW solution focuses on a built-in reasoning capability to discern the important information from the static. In battle, data can be received in realtime from a number of different kinds of sources such as: radar units, GPS devices, intelligence reports (sensor, electronic, and imagery), and other command and control systems. Making sense of this data is of crucial importance to saving lives on the field.

The resulting system used RDF as a way to store and manipulate instance data information. Standard RDF query interfaces provided realtime access to live data represented as triples. In many cases, but not all, these RDF triples were linked to OWL ontologies. The OWL information layer provided the ability to make inferences about data without prior knowledge about what kind of data would appear in the system.

Perhaps the single largest advantage that the Enterprise Semantic Web approach offered in this capacity was the ability to get reliable and flexible visibility into a wide range of data – without needing to effect a common data model. Typically, with other information visibility tools and techniques – such as EAI or EII – a common data model, or a series of "view" models are created for application access. The trouble with the common data model approach is two-fold: (1) the politics of creating common models is a significant barrier, and (2) common data models cannot adapt to changing schema.

For Network-Centric Warfare, the ESW offers a way to enable the information supply chain to adapt, reconfigure, and reason about new information in realtime. This prototype used a robust Service-Oriented Architecture (SOA) for the message-level interfaces, but leveraged the Enterprise Semantic Web for its information architecture, thus enabling systems to intelligently fuse information without requiring troublesome common data models.

Towards the Enterprise Semantic Web

In summary, the past thirty years of information-sharing technology has gotten us to the point where the most significant IT roadblocks we face are due to the sheer volume and complexity of digital information inside organizations.

The pessimists among us would say that technology innovation is dead, that we should just accept that today's time-tested methods (EAI & EII) for fusing data are the best we can do. They would also say that humans are always going to have to write lots of code to manipulate digital information and that we should just resign ourselves to a future vision rife with armies of coders working with tried-and-true languages like Java and C++.

Optimists will point toward the W3C Semantic Web activity as proof that a better future is on the horizon. They would point out that newer data representation technology will allow software developers to work with models more, and code less. They will say that the future consists of systems of systems that can adapt and respond to changing organizational environments, that the days of business knowledge locked in pre-compiled algorithms are numbered.

Regardless of whether you are an optimist or pessimist in this regard, there is growing evidence that the W3C Semantic Web initiatives are gaining acceptance, proving value, and being deployed in enterprise organizations throughout the globe. Regardless of whether you think that the Enterprise Semantic Web will fundamentally alter the path of IT innovation in the coming years, in fact, the businesses and government agencies who are adopting the technology believe that it will offer them greater flexibility, lower costs, and far more intelligent information systems.

About the Author

Mr. Pollock is a technology leader and author of "Adaptive Information: Improving Business Through Semantic Interoperability, Grid Computing, and Enterprise Integration." (John Wiley & Sons, 2004) As VP of Technology at Network Inference, Mr. Pollock is responsible for product strategy and envisioning next-generation adaptive enterprise software. Previously, as Chief Technology Officer of Modulant, he delivered one of the industry's first market-ready semantic integration middleware platforms in 2001. Throughout his career, he has architected, designed, and built application server/middleware solutions for Fortune 500 and US Government clients. Prior to Modulant, Mr. Pollock was a Principal Engineer with Modem Media and Senior Architect with Ernst & Young's Center for Technology Enablement. He is also a frequent speaker at industry conferences, author for industry journals, active member of W3C and OASIS, and engineering instructor with University of California at Berkeley's Extension on the subjects of Object-Oriented systems, software development process and software systems architecture.