Adaptive Information

IMPROVING BUSINESS THROUGH SEMANTIC INTEROPERABILITY, GRID COMPUTING & ENTERPRISE INTEGRATION

AVAILABLE FALL 2004 FROM JOHN WILEY & SONS

Authored By: Jeffrey Pollock
VP Technology, Network Inference

Ralph Hodgson
Executive Partner, TopQuadrant

Special Forward By: Patrick Gannon
President and CEO, OASIS

This book is about emerging technology that will enable a new kind of infrastructure. This new infrastructure is dynamic and adaptive – it configures itself and responds to change with little or no human input. No, we are not talking about magic adapters or artificial intelligence. This is an infrastructure driven by models of information, process and service interfaces. This infrastructure uses models to learn new behavior, find patterns and rebuild itself for different platforms. It is an infrastructure that can transform data on-the-fly, establish new communications without programming and follow process rules without knowing about specific services in advance. No, it is not magic – but it *is* smarter.

What You Will Learn in This Book

- ➔ Why semantics matter in IT
- ➔ Core middleware/vendor problems
- ➔ Vision of the future – a convergence of approaches
- ➔ The sound science that underlies new capabilities
- ➔ Semantic conflict patterns that cause brittle fixes
- ➔ How meta-data fits within next-generation solutions
- ➔ Emerging standards that will shape the future
- ➔ Patterns for the creation and use of ontology
- ➔ Architectures for building adaptive middleware
- ➔ Infrastructures for enterprise-scale solutions
- ➔ Case studies from companies who are doing it
- ➔ Adoption strategies & readiness planning

Smarter infrastructure is good for business. Business benefits achieved through the use of semantic interoperability, grid computing, and integration technologies described in this book include:

Increased Profits
- o Faster response to market demands
- o Better decision making
- o Cheaper adoption of new strategies

New Capabilities
- o Dynamic information repurposing and reconfiguration
- o Adaptive service-oriented networks
- o Loosely-coupled application connections and information

Reduced Costs
- o Faster data analysis and dissemination
- o Better quality of service in partner exchanges
- o Cheaper maintenance costs for IT systems integration

The authors of this book have watched several technology forces (such as semantics, agents, and services) align for many years. We have helped build software that reaches far beyond commonplace tools. We have interviewed technologists from across the globe so that we can report to you about the widest variety of techniques being leveraged to build this new kind of infrastructure. It is our firm belief that the collection of patterns, technologies and tools that we describe in this book will inevitably lead towards a future substantially better than today's software infrastructure realities.

Our goal in writing this book is to share the excitement of the vision that has become so clear to us, but also to provide useful patterns for understanding the scope and capabilities of the software driving this revolution.

Book Table of Contents

Advance Preview of Chapter 3: The Promise of Adaptive Information

Key Takeaways:

- ➔ Frictionless information flows are the inevitable future of information technology
- ➔ Models, independent of platforms and data, will drive semantic interoperability
- ➔ Component-driven service-oriented architectures (SOA) will provide flexible and dynamic connectivity – once they are fully enabled by semantic interoperability
- ➔ Autonomic computing concepts will drive strategic technology development in a number of industries and software solution spaces
- ➔ Intelligence information sharing problems are a useful context to examine the strengths of flexible, dynamic and loosely-coupled adaptive architectures

In an environment where business leaders freely discuss the end of IT innovation, it may seem overly optimistic to claim that semantics-aware middleware is the strategic future of technology, much less of business as a whole. However, it is true – the evolution of enterprise middleware is not only the strategic future of information technology, it also provides the single greatest potential for businesses to gain strategic differentiation through IT in the foreseeable future.

More common beliefs about what the future may bring include:

- A belief that many other technologies, not just middleware, will have strategic impact on the future of technology
- A belief that some other individual technology (eg: wireless, hand-held, or enterprise package software) is the strategic future of IT
- A belief that no technology can be strategic, because IT is only a tactical commodity

The central problem with either of the first two positions is that information sharing software, like middleware, is a trump card. Software that facilitates information sharing among physically or logically diverse enterprise applications is a critical path technology for every other enterprise information technology's success. Middleware is the enterprise cardiovascular system that carries an organization's lifeblood to many to its many parts. It is the path of least resistance to information contained in distributed enterprise applications such as: enterprise resource planning (ERP), customer relationship management (CRM), knowledge management (KM), network appliances, and supply-chain (SCM) radio frequency identification (RFID) applications. Middleware is responsible for connecting islands of information within the enterprise – thereby enabling other information technology components to reach its fullest potential. In short, middleware infrastructure is critical path for maximizing all other IT investments.

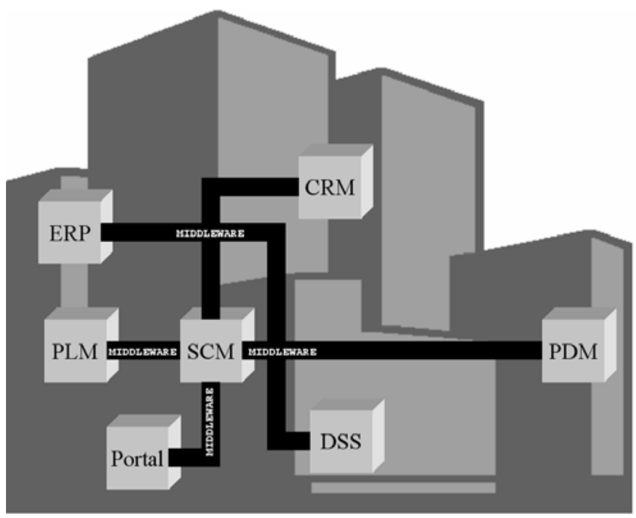

Figure 3.1: Software packages keep things running inside the enterprise

Middleware can be any technology that fulfills some significant portion of machine-to-machine collaboration, such as: CORBA, Web Services, EAI, B2B and custom-written solutions in the programming language de jour. Most middleware solutions currently in production are custom. Perhaps the most significant argument for middleware's position as the strategic future of software technology is the gap between what it does today and what technologists know it can do in the future. It is true that middleware's past failures leave plenty of room for doubt regarding software's strategic value to the enterprise. Skeptics use middleware's failed promises, and a myriad of other examples, to argue that IT cannot generate any strategic value at all.

The problems with the idea that IT is a tactical commodity and offers little, if any, strategic value are multifaceted.

- It assumes that all companies use technology the same way

- ➔ It assumes a static and level competitive playing field
- ➔ It assumes that the IT sector is highly mature and ignores the existence of disruptive technologies

For example, even though golfers have nearly equal access to high-performance clubs, their individual training and styles contribute to how well they use those clubs. Likewise, although information technology is a tool that everyone has access to, not everyone uses it in the same way achieving the same results. However, unlike the rules of golf, the rules of business change all the time – information technology can exploit new rules and provide advantages to savvy users who are on top of the game. Finally, and like Schumpter's economics, creative destruction is constantly generating higher quality, cheaper technologies to solve persistent business problems. The bottom line is that within traditional corporations and the IT industry as a whole – innovation marches on.

IT is indeed a strategic asset to innovative companies. The most strategically important part of IT in the coming years will be the adaptive and dynamic infrastructures enabled by semantics-aware middleware.

Organic Middleware: "Software, Integrate Thyself!"

We are on the cusp of significant changes in interoperability technology. These changes will drive substantial improvement in capabilities and value. Paradoxically, they will also reduce the complexity and expense of systems maintenance – though they will have far more complex underlying architectures and functions.

Various analysts, reporters, and business strategists have all seen these changes coming, calling it by different names – one of which has been the word "organic." Organic middleware means that the network will evolve, adapt, and change on its own, with minimal input from programmers, to make systems interoperable. This adaptability stimulus will drive efficiency, cost reduction, and emergent new behavior that can exploit new business opportunities.

Insider Insight –
Zvi Schreiber on Semantic Interoperability ROI

How do companies measure their return-on-investment for capturing the formal business meaning of their data? At Unicorn Solutions we introduce our customers to a methodology that measures value in four main ways:

- ➔ **IT Productivity** – Semantic tools can automate the generation and maintenance of data translation scripts. For an organization with many IT professionals, this creates a substantial and measurable ROI.
- ➔ **IT Efficiency** – Many organizations buckle under the cost of maintaining redundant databases and applications. Capturing data semantics based on an agreed ontology allows redundant systems to be decommissioned with measurable savings.
- ➔ **Business Agility** – Ontology provides a conceptual business view which may adapt to reflect business change quickly – this will result in quicker integration to new supply chain partners, quicker realization of savings after M&A, quicker optimization of business processes and quicker adoption of new products.
- ➔ **Information Quality** – Information quality problems, which according to TDWI cost US businesses $600 billion per year, are frequently the result of semantic inconsistencies among heterogeneous data sources.

Additionally, other soft benefits may include less dependence on specific vendors and their proprietary formats for capturing your data and business rules. Experience has shown that when real pain points are selected for semantic projects, payback takes a few months with massive ROI over the first year or two.

Zvi Schreiber
CEO, Unicorn Solutions

Table 3.1: Insider Insight – Zvi Schreiber, CEO, Unicorn Solutions

A Tough Road Forward

Middleware, and interoperability in particular, is one of few technologies that can offer rising returns on efficiency for existing enterprise systems. Unlike most packaged enterprise applications, middleware provides infrastructure and infrastructure provides foundation. But middleware has no face, no fancy GUI for every employee to log on to and check it out. Instead, middleware quietly hums away all of the time making sure that applications get the data they need at the right time.

At least that's how it should work.

The reality is that, just moving into its second decade of existence, information sharing software is a very young technology. Middleware solutions, both custom and off-the-shelf, tend to be inconsistently implemented with mixed results. Nagging fear, doubt and confusion plague IT buyers who doubt its ability to deliver on vendor's inflated promises.

Integration approaches today rely on vast amounts of human intervention even after initial implementation. Managing change has emerged as the most significant challenge in the information technology sector as a whole. Middleware fails miserably at achieving long-term savings and consumes vast proportions of annual corporate budgets with every change.

The fundamental technology barriers identified in Chapter 1 prevent today's approaches from achieving higher levels of success. Traditional middleware approaches discussed in Chapter 2 focus on the transport and orchestration of message delivery and not on its content – business information. Data is a second-class citizen in these common middleware integration frameworks. In order to enable truly organic and dynamic network capabilities a greater level of semantic awareness is a mandatory requirement. This new generation middleware must be built on much more sophisticated concepts of information and process to enable technology to drive strategic business advantage.

Organic Middleware Hinges On Semantic Interoperability

Data has always been the foundation of any information-sharing programs. Next generation systems rest upon an expanded view of data – information. In the figure below, interface level integration, method level integration, and process level integration have all developed on top of a foundation of data. With semantic interoperability, the expanded notion of data includes semantics and context, thereby turning data into information. This transition both broadens and deepens the foundation for all other integration approaches – enabling new organic capabilities to emerge.

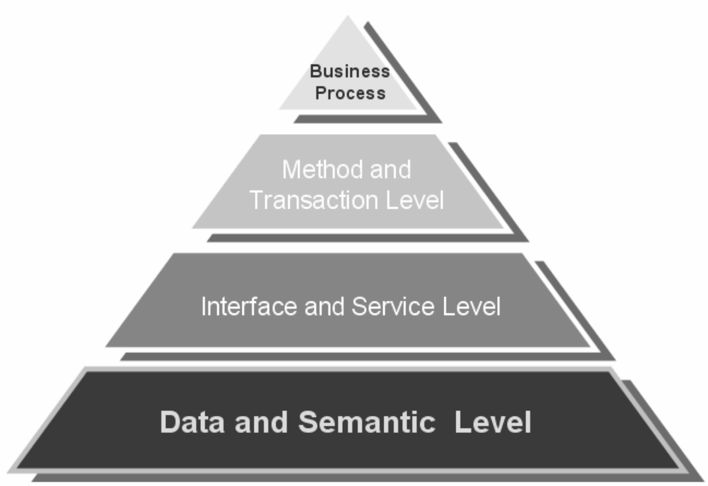

Figure 3.2: Hierarchy of integration levels

With a robust foundation of information, data and semantics as a baseline, new capabilities for interoperability can emerge:

- ➔ **Data Interoperability** – semantic interoperability of data enables data to maintain original meaning across multiple business contexts, data structures and across schema types by using data meaning as the basis for transformations
- ➔ **Process Interoperability** – semantic interoperability of process enables specific business processes to be expressed in terms of another by inferring meaning from the process models and contextual meta-data and applying it in a different process model elsewhere or outside the organization
- ➔ **Services/Interface Interoperability** – semantic interoperability of services enables a service to lookup, bind and meaningfully communicate with a new service without writing custom code in advance
- ➔ **Application Interoperability** – semantic interoperability of applications enable the granular interactions of methods, transactions and API calls between heterogeneous software applications to be platform independent
- ➔ **Taxonomy Interoperability** - semantic interoperability of taxonomy enables any category to be expressed in terms of other categories by leveraging the intended meaning behind the category definitions
- ➔ **Policy Interoperability** – semantic interoperability of policies and rules enables businesses to protect valuable resources regardless of what technologies their security mechanisms have been implemented in or how complex the rights management issues have become
- ➔ **Social Network Interoperability** – semantic interoperability of social networks enables people in different communities of interest to network, infer and discover meaningful connections through previously unknown contacts and interests

When these components can interoperate with either an automated or a human agent, network configuration can become emergent. Emergent behavior implies a level of dynamism and organic growth that can enable a pervasive autonomic environment with features like:

- ➔ Self-configuring interface and schema alignment across data vocabularies, directory taxonomy, service descriptions, and other components
- ➔ Self-optimizing transactions, routing, queries, and transformations
- ➔ Self-healing error flows that can recover from otherwise dead-end use cases
- ➔ Self-cleansing data validation that can scrub instance values from various sources

Fragmented Industry Efforts and Organic Computing

In spite of common goals and objectives, there are many competing and complementary efforts underway that all rely on innovative uses of data semantics. The following section will briefly introduce a few key technology movements that are driving momentum towards the future of semantic computing.

Autonomic Computing

IBM first popularized the notion of autonomic computing in its manifesto published in October of 2001[i]. This manifesto clearly established the needs and goals for the next evolution in information technology. Research and development of autonomic concepts has been a priority of IBM's research arm ever since and should ultimately lead to further innovation in IBM's software products. Microsoft has also embraced the autonomic concepts and applied them, not in middleware, but to data center technology. By late 2003 their public ad campaign of "automated systems recovery" was in full swing.

Definition – Autonomic Computing
Autonomic Computing[1] - Autonomic computing derives its name from the autonomic nervous system and denotes its ability to free our conscious brains from the burden of dealing with the vital, but lower-level, functions of the body. As used in the software industry, autonomic computing refers to self-managing systems that are comprised of four core characteristics: self-configuration, self-healing, self-optimizing, and self-protecting.

Table 3.2: Definition – Autonomic Computing

[i] For more information, see: www.research.ibm.com/autonomic/manifesto

More of a technology vision, autonomic computing doesn't necessarily represent any specific technologies or set of technical components. In fact, the visions of autonomic computing are somewhat different among some giants of software: IBM, Microsoft, Sun, and BEA. However, all of these companies are using technology to move towards more organic and change resilient infrastructures.

Semantic Web

The Semantic Web has become an umbrella term applied to everything related to processing semantics in data. Initially conceived of and articulated as an expansion of Internet capabilities to enable more meaningful searches among the vast amounts of data living on the Web, it has since morphed to include other notions of semantic processing on more traditional structured enterprise data as well. Indeed, the topics covered in this book – regarding the semantic interoperability of middleware and enterprise systems – are rightly considered part of the Semantic Web.

Definition – Semantic Web
Semantic Web[2] - The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation.

Table 3.3: Definition – Semantic Web

The World Wide Web Consortium, an international standards body directed by Tim Berners-Lee, controls the Semantic Web's vision and direction. Initial efforts with the Semantic Web have resulted in significant activity with knowledge representation markup languages (RDF & OWL) as well as the architectures and logic required to implement them alongside existing data sets. A wide range of companies, many are discussed in this book, have adopted the Semantic Web vision and are actively pursuing technology strategies that advance it further.

Semantic Web Services

Semantic Web Services are the intersection of the Semantic Web and Web Services. The Semantic Web Services Initiative (SWSI)[i], an international committee of visionary technologists, is defining how convergence will occur between these two technologies. A significant objective for the initiative is to create an automated and dynamic infrastructure for Web Services provisioning and usage. A secondary objective for the initiative is to assist in the coordination of research efforts in commercial and academic programs.

Definition – Semantic Web Services
Semantic Web Services[ii] - Semantic Web Services are a Web Service implementation that leverages the Web Ontology Language Service specification (OWL-S). OWL-S supplies Web service providers with a core set of markup language constructs for describing the properties and capabilities of their Web Services in unambiguous, computer-intepretable form. OWL-S markup of Web Services will facilitate the automation of Web service tasks including automated Web service discovery, execution, interoperation, composition and execution monitoring. Following the layered approach to markup language development, the current version of OWL-S builds on top of W3C's standard OWL.

Table 3.4: Definition – Semantic Web Services

Service Grid

A service grid is a concept represents a specific set of technology components. The Open Grid Service Infrastructure (OGSI) specification extends Web Services WSDL specifications to enable more dynamic usage patterns for service-oriented architectures. The Global Grid Forum (GGF) is a technical community process, also focused on processor-centric computing grids, that drives the OGSI and Open Grid Service Architecture (OGSA) technical specifications.

---

[i] For more information, see: www.swsi.org
[ii] Definition adapted from: www.daml.org/services

Definition – Service Grid
Service Grid[3] - A service grid is a distributed system framework based around one or more Grid service instances. Grid service instances are (potentially transient) services that conform to a set of conventions (expressed as WSDL interfaces) for such purposes as lifetime management, discovery of characteristics, notification and so forth. They provide for the controlled management of the distributed and often long-lived state that is commonly required in sophisticated distributed applications.

Table 3.5: Definition – Service Grid

Both IBM and Sun Microsystems have already made substantial investments in the grid-computing arena. They both participated in the specification process, each sponsoring their own proposed specifications for various portions of the grid computing challenge. With continued support, some combination of OGSA efforts and the Semantic Web Services efforts will deliver a standards-based approach that finally delivers on the initial promise of Web Services – dynamic service discovery.

Model-Driven Architecture

Originally focused more at the application level than at the enterprise network level, model-driven architecture (MDA) concepts and specifications will drive the next evolution of code generation for enterprise information systems. Consisting of platform-independent models and platform-specific models, the MDA insulates business and application logic from technology evolution. Using MDA, software engineers spend more time with enterprise models than with application code. A step beyond CASE (computer aided software engineering) technology, code generation is the foundation of MDA.

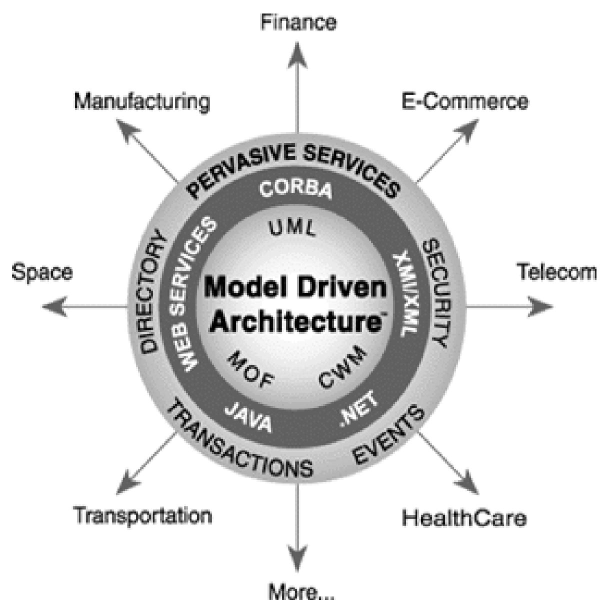

Figure 3.3: MDA vision depicted by the Object Management Group[i]

Definition – Model-Driven Architecture
Model Driven Architecture[4] - Model Driven Architecture is an approach to system development that emphasizes the use of models to separate the specification of software application independent from the platform that supports it. The three primary goals of the MDA are portability, interoperability, and reusability.

Table 3.6: Definition – Model Driven Architecture

[i] Image is property of Object Management Group, www.omg.org/mda

The Object Management Group (OMG), who have brought us many innovative standards (CORBA and UML) now offers the MDA. Adopted in 2001, the ongoing efforts of the OMG in this regard will continue to drive the industry toward a model-centric method of software development. More on the MDA is covered in Chapter 6, *Meta-Data Archetypes*.

Intelligent Agents

Agent technology has had a long, but anti-climatic, history. For nearly a decade, agent technology was supposed to have brought us all closer to IT nirvana. However, multiple barriers to agent capabilities have prevented this from happening. To begin with, there is no single definition of what an intelligent agent is. This book uses the definition provided by Dr. Nicholas Jennings, a leading researcher in agent technology.

Definition – Intelligent Agent
Intelligent Agent[5] - An agent is an encapsulated computer system that is situated in some environment and that is capable of flexible, autonomous action in that environment in order to meet its design objectives.

Table 3.7: Definition – Service Grid

Agent technology will not offer widespread benefits to the enterprise in and of itself. Instead, it requires an infrastructure to support agent activities. When coupled with other technologies, like service grids, model-driven architecture, and inference engines, agent technology will become a crucial aspect enabling dynamic, adaptive, and autonomic enterprise computing.

Achieving Synthesis

Each of these previously mentioned technology innovations would accomplish significant new breakthroughs in IT on their own. But the synthesis of these approaches is what will drive momentum towards a holistic, dynamic, and organic middleware framework. Likewise, each of these technology innovations relies on an improved information framework that enables the full scope of their capabilities.

Let's take a look at how the synthesis of these technologies might be applied in a hypothetical scenario involving security and intelligence information sharing.

Intelligence Information Sharing in the 21st Century

Homeland defense is a pressing problem that highlights many of the barriers faced by current information interoperability software. Non-technical issues such as organizational culture barriers, decentralized IT management, politics of negotiation and the sheer complexity of policy and change management are quite significant across federal government agencies. These challenges are extreme examples of what all information sharing programs face – albeit in lesser scale and with fewer worst-case consequences. The scope of these challenges for intelligence information sharing and homeland defense offer a poignant scenario for examining some of the requirements that next-generation middleware should meet and points towards the kind of technical architecture needed to solve them.

The remainder of this chapter will explore these intelligence information sharing problems and explore high-level technology capabilities that may provide a solution. As you read the following scenario, remember that the synthesis of emerging technologies will be greater than the sum of its parts. This example highlights only the portions of the overall architecture that depend on semantic interoperability for information sharing.

Information Sharing Imperative

While impossible to pinpoint the precise beginnings of an information sharing network for the United States, Paul Revere's ride in the spring of 1775 marked one of the earliest examples. From the famous cries of, "The British are coming! The British are coming!" the demand was already apparent for an efficient mechanism to relay information when national security is threatened.

Today, much has changed. We no longer need to hang lanterns in church windows and ride swift steeds to pass emergency messages among our communities. Computer networks and telephones shrink even the greatest of distances.

However, much is also the same. In spite of significant advances, computer technologies are only 30-50 years old, which make them immature compared to other modern technical luxuries, and they are generally incapable of performing even the most basic communications without being assisted by skilled human gurus.

To show how important better information sharing is, consider a report by the Markle Foundation Task Force on National Security in the Information Age entitled "Protecting America's Freedom in the Information Age," which provides a sobering analysis of the dangers we face.

It is impossible to overstate the importance of swift, efficient, and secure information sharing to our local, state, and federal agencies. However, to state that this challenge has been difficult to accomplish is an understatement.

Information Sharing Conundrum

A seemingly simple task, information sharing has proven to be a treacherous sinkhole of time and money for our government. And lest you think that these woes are the fault of bureaucracy, it is important to note that large business enterprises routinely fail at the same task – upwards of 88% according to some analysts[i].

The reasons for our problem with achieving swift and efficient information sharing in homeland defense are intricate and intertwined. These problems also cast a bright spotlight some issues that are usually seen as minor problems in commercial enterprises, but have enormous consequences.

- ➔ **Technical Incompatibility** – usually the most cited problem, compatibility issues tend to be black and white in nature. Typical compatibility issues will boil down to application protocol, programming language, or data definition problems that can be fixed, but sometimes only at significant cost.

- ➔ **Operational Limitations** – in the no-so-distant past this has perhaps been the most significant barrier to joint efforts on information sharing. Differences in agency objectives have led to differences in processes, equipment, and organizational authority. Along with cultural issues operational limitations have greatly prevented substantial progress.

- ➔ **Budgetary Constraints** – the nature of budget constraints has been two-fold. Both inadequate funding and the lack of specific plans for coordinating different programs have limited the possibilities of finding ways to share required information.

- ➔ **Cultural Differences** – often overlooked, this is frequently the difficult to spot because it is not a technical problem. Agency culture is embedded in the way one government agency chooses to operate or the way differences in terminology evolve among organizations. Culture is subtle, culture is folklore, and culture is pride in differences – which often leads to difficulty when sharing information. Considering the historical differences between the State Department and the Defense Department, various agencies in the intelligence community (for example, CIA, NSA, DIA and other intelligence agencies), and agencies tasked with overseeing the nation's borders (for example, U.S. Customs, Border Patrol, etc.) can highlight examples of differences along these lines.

- ➔ **Political Wrangling** – like it or not, people in power often wield that power in order to secure even more. Information is power. This persistent power struggle leads to fiefdoms of control and dangerous silos of isolated information. Even where power is not the prime motivation, many

[i] Gartner Group Report, December 2001

agency leaders worry that ceding operational control of information could compromise their primary mission of their agency – this mentality leads to information hoarding.

It is not impossible to overcome these challenges. As the Markle Task Force working group "Connecting for Security" has suggested, there are at least 10 major characteristics that the next-generation homeland security information network should meet. In addition, the Federal Enterprise Architecture (FEA) Working Group has specified a minimum set of frameworks that should unify agency development programs. Other groups, such as the Regional Organized Crime Information Center (ROCIC) have installed information sharing systems that mostly consist of Internet portal type applications (such as RISSnet) that enable specialists to search by keywords within various systems for data.

Perspectives – Markle Key Characteristics:[6]

- ➔ **Empowerment of Local Participants** – empowerment to access, use, and analyze data on the network's edge
- ➔ **Funding and Coordination** – decentralized, but adequately funded, coordinated, and empowered participants
- ➔ **Safeguards to Protect Civil Liberties** – guidelines should be implemented to prevent abuse
- ➔ **Elimination of Data Dead Ends** – stovepiped data should not lead to a dead-end, with no accountable authority
- ➔ **Robust System Design** – Minimize possible points of failure
- ➔ **Network Analysis and Optimization Capabilities** – detecting patterns in the security network may trigger alerts
- ➔ **Solid Growth and Upgrade Plan** – the use of good architectural patterns ensures maximum longevity
- ➔ **Support for Existing (legacy) Infrastructures** – a successful network will exploit existing information resources
- ➔ **Create Network Aware Scenarios** – complex models of security scenarios can assist analysts in the field
- ➔ **A Connected Culture** – being connected and participating must become second-nature to participants

Table 3.8: Perspectives – Markle Foundation homeland defense characteristics

There are two primary demands for information sharing: (1) enable visibility into disparate systems, preferably with analytic capabilities to identify trends that develop over time and across systems and (2) to enable interoperability among these systems so that local alert systems and agency software applications can operate on that data directly, from within their own environments. Early steps have been taken to accomplish both of these goals, but mostly by uncoordinated smaller efforts with little cross-mission planning.

This problem, when viewed in entirety, is exceedingly complex. Federal agencies already have volumes of information that can legally be shared, so interagency exchange is technically possible – in fact, most consulting companies would be happy to work on specific inter-system communications. The core challenge today is to imagine an infrastructure that can be built, and built upon, in the years to come – while enabling government agencies and commercial entities access to required security information on demand.

Given enough money and technology staff, any software vendor or systems integrator could propose a reasonable plan. However, the future of such an infrastructure must also account for the human factors.

Human elements are frequently assumed to be a non-reducible factor of technology deployments. Technologies for middleware and information sharing lack contextual sensitivity in their deployment architectures – which leads to custom-written code that solves the same problems over and over again. No matter vendor-supplied tools an end-user works with, the result of software deployments inevitably requires that a programmer be proficient in Java, XSL/T, XML, SQL, or some other language in order to manipulate and give meaning to the raw data moving through the system.

Normally this is not a problem. But consider the scale of a broad-based cross-agency intelligence sharing solution. Potentially hundreds of thousands of unique data sources may need to contribute or tap into relevant information. This is a middleware solution of a scale that has never been considered before. It is a middleware problem that, based on person-hours alone, could never be solved with today's technologies.

Today, software's connectivity and information architecture concerns are mixed into the same physical components – creating tightly coupled data and communication protocols. In environments that change frequently this brittleness results in expensive maintenance scenarios.

To reiterate, these issues have been overcome in the past because most middleware ties together dozens (not hundreds of thousands) of applications – and human-power is much more cost effective and manageable at those levels.

In summary, there is much agreement on the core functional characteristics that a successful solution must meet. The human factor, and scalability of possible solutions must be a key consideration to find a reasonable answer. Unfortunately, popular notions about integration technology – and the vast amount of disinformation that surround commonly available high-profile vendor offerings – confound these goals.

Towards a Pragmatic Solution: Semantic Interoperability

Thinking about a solution for a problem of this scope and this complexity is extremely challenging. To frame it in terms that are policy-oriented or business focused would result in a solution that fails to meet fairly specific technical goals. To frame the solution in terms that are technology focused will result in tactical solutions that are missing the forest for the trees. A balance of these forces is required.

A balanced solution will enable us accomplish both the technical and non-technical requirements of an acceptable solution. These requirements derive from the characteristics and goals put forth by the Markle Foundation task force and succeed at addressing problems at multiple levels such as cultural, political, technological, operational, and budgetary.

Due to the need to balance diverse and loosely defined concerns while still addressing multiple layers of complexity, it is pretty clear that what we need from technology in the future is vastly different that what is generally available today. We will need it on a scale that has not yet been seen before. We will require it as a matter of safety – not just for generating return on investment. We will need to apply new thinking to old problems in order to determine an ideal course of action.

We start with a few core premises:

- ➔ For any solution to succeed, a high degree of organizational autonomy among federal agencies should be supported. In other words: few, if any, vendor-based, proprietary or narrow vocabulary standards should be required.
- ➔ Organizations must be able to support a wide variety of historical, legacy, applications and processes. It is not practical to assume any baseline of technical competency when planning for the infrastructure.
- ➔ The solution must be infrastructure, which is supported by appropriate tools, methodologies, and verification to allow for non-deterministic deployment options. (it must be self contained and relatively easy to maintain and deploy in a decentralized operating environment)

A fundamental aspect of a successful solution is to apply the architectural pattern of separating concerns. In this proposed architecture data semantics are purposefully separated from the connectivity technologies as a way to divide and conquer the inherent complexity of the system.

Some publications[7] have referred to the distinction between logical and physical components of middleware. For the purposes of this solution, the logical layer encompasses logical data models, conceptual models, business rules, process models, and logical query structures. The physical layer traditionally encompasses connectivity protocols, workflow brokers that execute process and routing models, data persistence mechanisms, data exchange formats (instance data), physical security rules, and various tools to manage systems participating in exchanges.

Semantic Interoperability Framework Characteristics

One way to describe a system is with a set of buzzwords. A standard set of them has been used to describe the framework. The rest of this section is to explain what is meant by those buzzwords and the problems that are being addressed.

Definition – Semantic Interoperability Framework
Semantic Interoperability Framework – A highly dynamic, adaptable, loosely-coupled, flexible, real-time, secure and open infrastructure service to facilitate a more automated information sharing framework among diverse organizational environments.

Table 3.9: Definition – Semantic Interoperability Framework

Dynamic

The essence of being dynamic hinges upon the infrastructure's ability to support the configuration of information assets in previously unforeseen ways without the intervention of humans. To say it another way, sometimes an interaction between two sources of information needs to take place, but development teams had not predicted that particular interaction, therefore no code had been written to facilitate the transaction. This leaves the organization with an option to either expend development person-hours to get the data required (if they are even aware of the request at all) or to do without the information.

The semantic interoperability framework makes use of three core technologies that enhance the dynamic nature of the final solution and reduce the need for human intervention:

- ➔ **Dynamic Data Transformations** – by creating an architecture whereby data schemas are linked via context-sensitive maps, the underlying runtime components can generate the transformation code on the fly – without having to have prior knowledge of the specific interaction desired.

- ➔ **Dynamic Query Generation** – by providing rich conceptual models (ontology) that intelligent agents and/or dedicated inference engines can navigate with the aid of description logics, the software can dynamically assert truth statements in the form of queries against previously unknown schema types.

- ➔ **Dynamic Map Generation** - by providing human users, or intelligent agents, the ability to assign relationships among concepts basic semantic equivalence can be asserted among schema entities, thereby enabling the software to make intelligent guesses at relationships among schema types.

- ➔ **Dynamic Discovery of Information and Services** – by providing agents and humans the ability to lookup new items on registries or via taxonomy. This is a fairly well understood capability that has driven much of the initial excitement around Web Services. While not trivial, we are much farther along in this area of dynamic capabilities than the others.

These dynamic capabilities are the key enablers for eliminating the low-level involvement of programmers and application experts in the day-to-day operations of information sharing. The ability to initiate and complete round-trip transactions and intelligence operations on-the-fly and without prior knowledge about specific outcomes is essential to a sustainable long-term vision of information sharing at the scale we require for homeland security initiatives.

Real Time

People coordinating the day-to-day requirements for information sharing should be able to establish previously unforeseen data sharing requirements rapidly, without months of development time, without months of analysis, and without the costs associated with all of that effort. Real-time is the capability of establishing new modes of communication among services and systems on demand and in real-time.

Of course the more technical meaning of the term "real-time" must also be a core capability: to facilitate transactional message exchanges over different network topologies (ie: publish-and-subscribe and request/reply) without significant latency. Closely associated with real-time messaging is queuing technology – resulting in guaranteed delivery for mission-critical data delivery when lives depend on it.

The bottom line is speed. Speed to respond to new requirements, and speed to get the data to the right place.

Loosely-Coupled

A basic tenet of systems design and architecture is indirection. Indirection is a concept that is used to plan for future uncertainty. Simply put, indirection is when two things need to be coupled, but instead of coupling them directly, a third thing is used to mediate direct, brittle connections between them. A simple example of indirection is one that you probably use everyday. When you type a web address, such as www.yahoo.com, into your web browser it will not go to yahoo directly. First it is directed toward a DNS (domain name server) that tells your computer what IP (internet protocol) address the simple name www.yahoo.com is mapped to. These numeric addresses, like phone numbers, identify the correct location of the yahoo website.

Indirection is used to insulate separate concerns from being too tightly connected with each other. For instance, when yahoo changes their IP address, which they may do all the time, your computer system doesn't break when you try and connect to it – because of indirection via the DNS.

In object oriented software systems, indirection is used in software patterns such as proxy, façade, and factory, among others. In a semantic framework indirection is a fundamental aspect of ensuring loose-coupling for the purposes of efficient, dynamic, and automated information sharing.

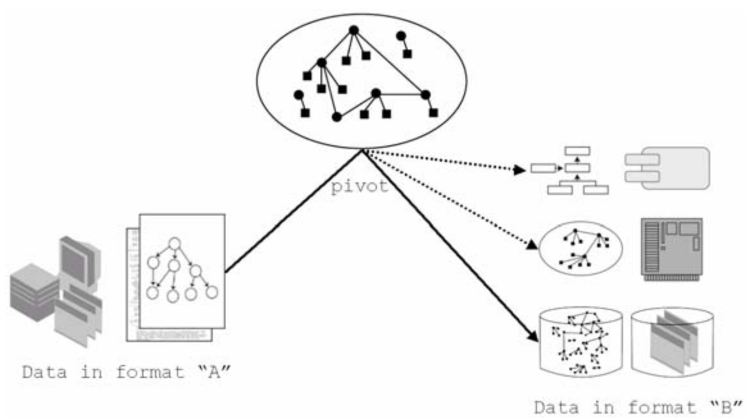

Figure 3.4: Indirection with data schema and ontology

Abstraction is also a primary enabler for this kind of architecture. If you could imagine the above scenario of using a pivot data model without abstraction, it would require the aggregation of all of the data elements in a particular community. The result would be a community of 500+ applications, each application with approximately 100 data elements, requiring a pivot model with about 50,000 data elements. An abstracted model would conceivably be capable of representing this information in far fewer than 100 data elements.

By creating loose-coupling in the fundamental aspects of the technology, the semantic interoperability infrastructure is built for change. It's this built-in capability to manage change, adapt, and to be flexible that makes this infrastructure concept more powerful than other approaches.

Highly-Flexible

Each enterprise system or information resource should be capable of participating in configurable and adaptable interoperability architectures – despite different core connectivity platforms.

A semantic interoperability framework must be pluggable. Pluggability means that different core technology platforms can make use of it by simply connecting and publishing some basic meta-data about itself. Functionally speaking, the framework should be able to support many different kinds of information technology resources. Resources such as:

- ➔ Applications (software that provides functions to users)
- ➔ Middleware (software that connects other software)
- ➔ Databases (software that stores data)
- ➔ Portals/Hubs (software that distributes functions broadly)

Flexibility also implies an ability to work with various kinds of technical transaction modes such as publish-and-subscribe events as well as request/reply calls. These transaction modes often define middleware deployment topologies – at the network and message levels. Typical deployment topology include:

- ➔ **Hub-and-Spoke** – first popularized with early deployments of NEON middleware, the hub-and-spoke deployment architecture is textbook request/reply message transactions typical of many popular vendor offerings

- ➔ **Message Bus** – the message bus architecture is usually employed with a publish/subscribe event management scheme, this architecture was popularized and most frequently associated with Tibco middleware offerings

- ➔ **Web Services Grid** – the new-comer on the block, the Web Services grid is the registration of many application and utility interfaces and service descriptions in an open network cloud. This allows other members to use their services dynamically; so long at it is consistent with appropriate security policies.

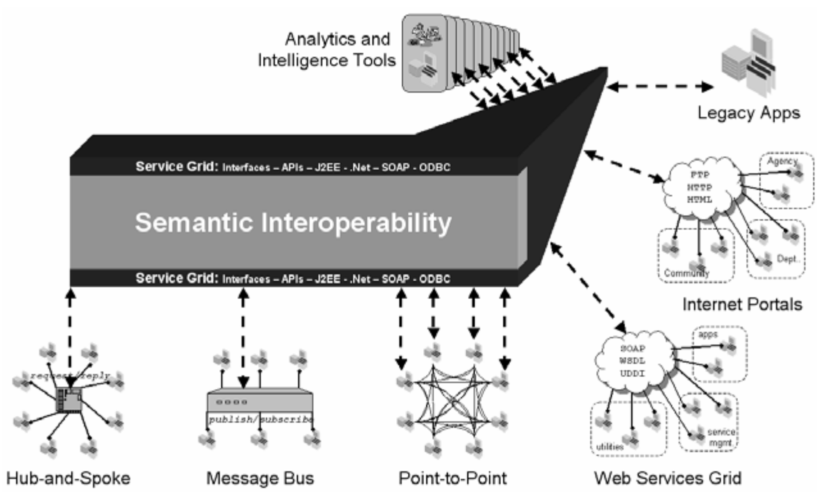

Figure 3.5: Flexible accessibility from enterprise architectures

The framework needs to unify data across diverse topology ecosystems by making data of all types – including unstructured, semi-structured, and structured – interoperable. The semantic interoperability framework makes middleware architectures more flexible by embedding machine-processable meta-data required for interpreting the meaning of operational data.

In today's technical terms, the framework must also be accessible via industry standard APIs including Web Services, Java and J2EE, COM and DCOM, CORBA, and others.

Secure

Security, like semantics, is a technical component that is difficult to draw a single box around. In fact, security needs exist at every layer of a complete technology architecture and is inherent in the tools we use during deployment. Network security is a different kind of mechanism than application security – and application security is a different kind of security than data security.

At the network level, mechanisms such as network firewalls are part of a successful infrastructure. At the application level, the application firewall is the cornerstone tool that provides security to the deployed information framework. At the data level, mechanisms should exist that separate different realms and hierarchical access points to sensitive information. Various models of access levels can be applied from the CIA, FBI, law enforcement, and the department of defense.

Without going into excessive details here, common information technology security components would typically include:

- ➔ Firewalls
- ➔ Application/network authentication/authorization
- ➔ Private/public key encryption
- ➔ Malicious attack protection
- ➔ Certificates and signature support
- ➔ Encryption and Hash
- ➔ Auditing and reconciliation

The semantic interoperability framework should make use of each of these proven, industrial strength security precautions. Most importantly, this entire proposal is about security – helping us gain confidence in our nation's ability to respond before and after potential threats to its citizens.

Open

A solution built to stand the test of time will meet the criteria of the openness test:

1. A solution should not depend on proprietary products or fail to accommodate open standards in order to best meet the test for openness.

    - ➔ Platform Neutral Services (web service, J2EE, .net)
    - ➔ Schemas (OWL, RDF, UML, Express, XML)
    - ➔ Registries (UDDI, LDAP)
    - ➔ Communication Protocols (HTTP, RMI, SOAP, FTP, SNMP)

2. A solution should remain neutral with regard to core business differences to appropriately meet the test for openness.

    - ➔ Data vocabulary mediation – agree to disagree
    - ➔ Query mediation – write once, query everywhere
    - ➔ Process mediation – loosely-coupled process sharing

3. A solution should not require information consumers to use the brand of application software or middleware that the information provider uses in order to best meet the openness test.

    - ➔ No adapters required
    - ➔ Insulate the business process from other organizations
    - ➔ Insulate the internal technology from other infrastructures

Service-Oriented

A robust information framework is an on the wire service that is always waiting to be used. It is a non-functional (in terms of providing business functions to end users) part of the overall systems infrastructure providing a utility to the rest of the applications, systems, and processes within the security community.

The infrastructure service ensures that applications and processes can get on the grid and plug into a broad service utility. It is the kind of software component that gets built, and then forgotten about. Once in place, the services continue to be used, but the malleable aspect of the solution is maintained through the schemas, business rules, and domain models registered within the framework.

Information Centric

Throughout the history of computing, information has taken a backseat[8] to other computing aspects. Whether code, messages or processes – it seems that people just want to assume that data will take care of itself. This lack of information-centricity has created a massive problem with the way data is stored and represented. The semantic interoperability framework challenges this central problem through recently emerging approaches for working with data semantics. Three core principles are applied to a computing environment to formalize the description, management, and interoperability of information assets: mediation, mapping, inference and context.

By separating the management of information from processes and software applications – through the use of context – a pluggable environment may be created that dramatically reduces new solution's development time and excessive maintenance costs. In this way, the semantic interoperability framework can become the chassis for the next-generation of government information technology.

Autonomic

Solutions that have a high degree of human involvement, due to a lack of automation, experience much higher costs, much longer cycle times, and higher maintenance. In the integration world it has been notoriously difficult to automate key processes of linking systems together. Due to different technologies, cultures, and organizational processes specialized resources are almost always required to be hands-on when integrating systems. In fact, a great deal of integration is still done with "swivel-chair" methods whereby people literally type information from one system to another.

For a next-generation solution to change the rules of how involved people need to be in creating middleware solutions, we need to introduce new technologies that can learn more about businesses. The framework should leverage these technologies to automate the process of several key areas of middleware solution development:

- ➔ **Query Generation** – rules and constraints associated with querying databases, XML documents, and custom APIs.
- ➔ **Data Transformation** – algorithms and structures for manipulating instance data in a variety of forms
- ➔ **Schema Mapping** – relationships and entity associations among a community of related schema
- ➔ **Schema Generation** – structure and concepts of information in both broad domains and specialized applications, for unstructured, semi-structured, and structured data

Part of the overall goal of automation is to build in capabilities for the framework to evolve on its own. Steps in this direction have been partially implemented in other systems, for example: data mapping tools use thesauri to associate synonyms to data elements – these thesauri can store new associations that help to further refine the semantics of the thesauri each time a new map is completed. Other capabilities that exhibit elements of evolutionary growth can be found in the commonly used ranking systems employed by websites such as Bay, where community participants rank the value of the information and services they receive from others.

Next generation tools will incorporate these kinds of ideas to form a type artificial intelligence (AI) within middleware. We will see these capabilities highlighted by software that can infer data meanings, alter data semantics dynamically over time, establish community rules for interactions, and reason about the information and services that are widely available. These strengths will ensure that humans are not required to facilitate the programming of rules at the very lowest levels – and thereby ensure that we can more rapidly make use of the information distributed across agencies, organizations, and commercial entities.

Developing a Semantic Interoperability Solution Architecture

It's no secret that a solution with all the characteristics presented in the previous section does not yet exist. However, tomorrow's semantic interoperability capabilities will fulfill these promises with a combination of several technologies that are already available.

Tomorrow's information sharing platforms will contain key technologies that enable loosely-coupled, dynamic, and more automated information sharing:

- ➔ **Ontology Tools** – tools that create, manage, and link conceptual models, taxonomy, and canonical models to actual enterprise data, process, and business rules schemas.

- ➔ **Dynamic Mediation Tools** – tools that operationalize the ontologies within or across organizations to mediate conflicting processes, policies, data and business rules without custom-written programs for the purposes of information sharing.

- ➔ **Inference Tools** – tools that dynamically navigate ontology and schema to facilitate the generation of new schemas, queries, business rules in a loosely-coupled model-driven environment.

- ➔ **Thesaurus Tools** – tools that allow non-expert information consumers to identify basic semantics for data, processes, and business rules – thereby enabling automated relationship building among schema, ontology, and business rules.

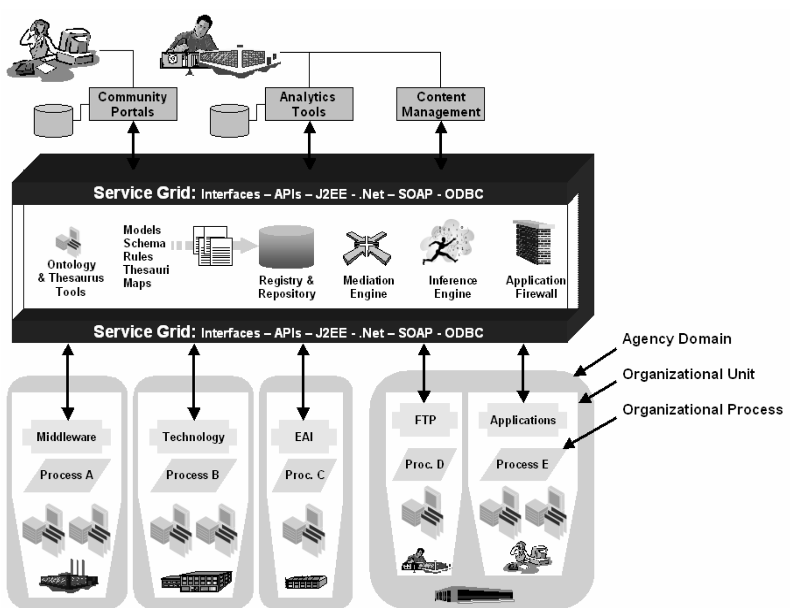

Figure 3.6: Plugging into the semantic framework across agency domains

These four core information sharing mechanisms will complement many of the now standard protocols and data sharing techniques that are commonly understood, such as:

- ➔ Web Services
- ➔ EAI/B2B/MOM Messaging
- ➔ J2EE/DCOM APIs
- ➔ Application and Network Security
- ➔ Registry and Repository Services
- ➔ Other communications protocols

With the proper architecture utilizing these core information sharing toolsets, a wide array of flexible interfaces, and utilizing a number of possible network data exchange topologies – a complete solution may be constructed.

Design Time

One way to differentiate the views of the architecture is to consider the differences inherent in design time activities vs. run time activities.

Design time activities are those that are typically performed prior to any physical attempt to connect, integrate, or otherwise make interoperable a set of two or more applications.

Typical activities done in this part of an integration lifecycle include:

- ➔ Requirements Analysis
- ➔ Object Modeling
- ➔ Logical Data Modeling
- ➔ Conceptual Modeling
- ➔ Business Rules Identification
- ➔ Data Dictionaries and Thesaurus
- ➔ Workflow and Process Analysis
- ➔ Interface API Descriptions

As the figure below indicates, the intent of semantic framework is to make the results of all of this design-time work storable and executable within a shared platform.

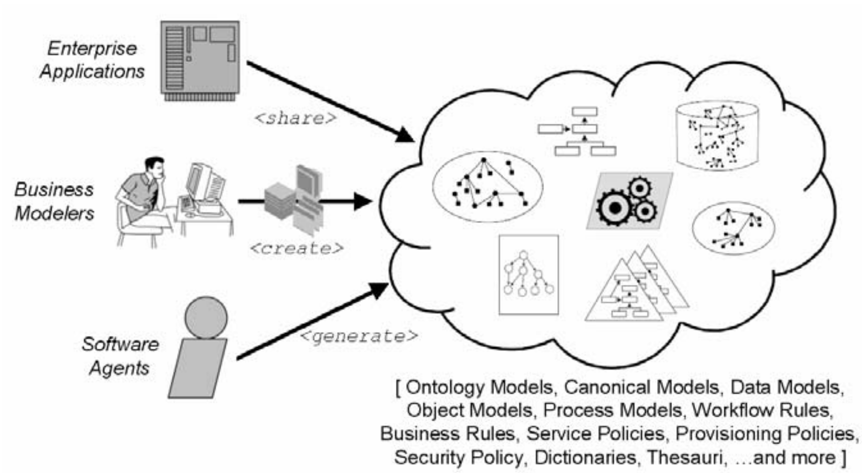

Figure 3.7: Design time processes and artifacts

Once we have the opportunity to drive the run time processes from the actual artifacts that are created during the design time portion of a project, many efficiencies will be realized. These efficiencies that are created will translate into fewer person-hours spent in the design and development phases of integration efforts. Traditionally, programmers are asked to interpret the design intent of numerous design deliverables into executable code. Instead, the design deliverables (models, workflow, business rules, dictionaries, etc.) will be made available to the run time layer directly, to enable the dynamic configuration of a wide variety of possible data and process interactions – without attempting to predict the precise nature of a data or process exchanges.

Run Time

The persistent challenge to integration has been the amount of human involvement required to get data, processes, and systems that have been developed at different times, in different places and by different people to interoperate. The subtleties in business rules and semantics associated with these challenges (data, processes, and systems) have always been more of a gating factor than any specific technical issue of connecting to a new application. It is these factors that mandate significant human involvement in the process of software development. Software development processes, guide engineers through a formalized process of understanding and analyzing a problem, then designing, developing, and testing a solution for it.

This is one core reason why today it will take months, instead of minutes, to make one set of databases at a local sheriffs office to communicate with legacy applications maintained by an international crime database.

An adaptive software infrastructure would change the rules of the game.

Semantic interoperability succeeds because it reduces the amount of human intervention in the acquisition of data from previously unknown data sources. The premise of the framework's runtime is to accept any data, any time, and over any protocol. Runtime service components would be responsible for deriving the meaning of data. Once the meaning was established in a platform neutral way, it would make the data query-able and transformable into the context of any other system that is also plugged into the framework.

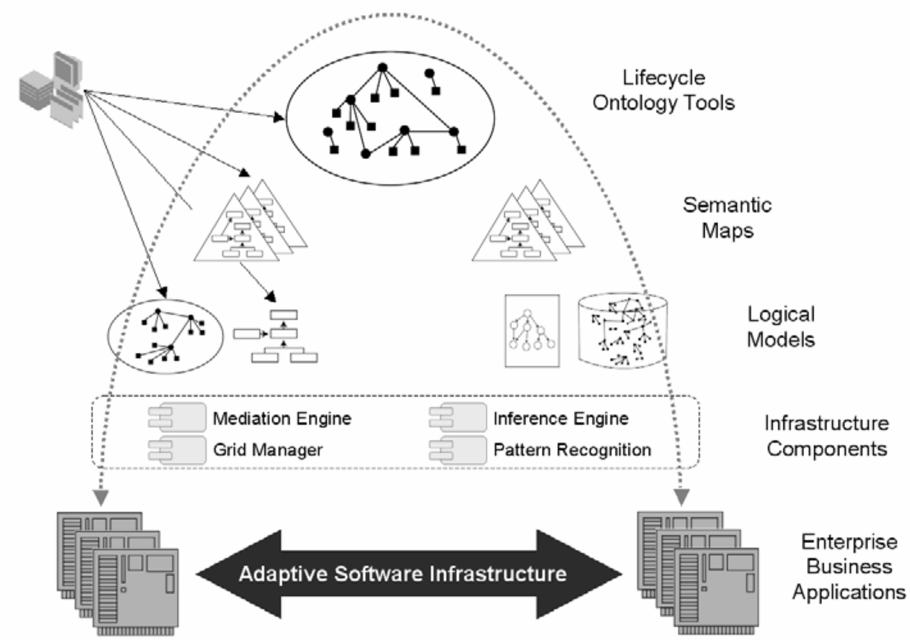

Figure 3.8: Any to any, dynamically, without apriori knowledge custom code

If we start with the baseline assumption that a source will have access to the framework's architecture over a communications channel – which can vary by application or security requirements – then the process of sharing information begins with registering process and data models. This process can be automated in ways that reach far beyond what we presently do when designing and writing programs to accomplish tasks – and provide a multitude of benefits over time that custom written code cannot.

Once these steps were completed, the framework would have enough "knowledge" about the data, process, and business rules to dynamically query, transform, and share the local application's contents with any other framework subscribers without having to predict the precise nature of interactions in advance.

Of course, this scenario is a simplification for the purposes of presenting a high-level flow of events, some non-trivial assumptions include:

- ➔ the existence of one or more community models (process or data) that are sufficiently abstract to serve the purpose of a mediating schema
- ➔ the existence of tools to infer popular community usages and help direct the appropriate mapping into the common (pivot) formats

- ➔ the linkages of diverse local query syntax to a common neutral syntax that operates on ontology (ie: the mechanisms to map description logics to traditional query interfaces)

So, while this architecture may sound complex and uninviting today, the benefits of what it delivers far outweigh the trepidation of attempting to use it to solve our pressing problems. There are no other possible solutions, save humans coding queries, transformations, and business rules over many months of development time.

Final Thoughts on Frictionless Information

Although we used homeland defense as an illustrative example, the issues and requirements in commercial businesses closely mirror the demands of agency-to-agency interactions. As difficult as it may be to admit, a ready-made out-of-the-box solution for this kind of advanced information sharing does not exist to day. What we have been describing in this chapter is still largely just a vision. However, the next-generation of tools that will meet advanced intelligence sharing requirements do exist – some assembly required!

Fortunately we are in a strong position to draw on expertise from several key disciplines to enable this convergence occur. Experts in the fields of ontology engineering, Semantic Web, Web Services, EAI/B2B, inference engines, data and schema mapping, workflow and process modeling are guiding the next generation of information sharing platforms each and every day. Part Two of this book, The Semantic Interoperability Primer, focuses more deeply on the nature of these emerging technology programs and begins to identify some patterns of their usage.

---

[1] The Dawning of the Autonomic Computing Era, A.G. Ganek. IBM Systems Journal, Vol 42, NO 1. 2003.
[2] The Semantic Web, Berners-Lee, Hendler, Lassila. May 2001
[3] Grid Service Specification, Draft 3 2002. The Globus Alliance
[4] MDA Technical Guide Version 1.0. Object Management Group, 2003
[5] N.R. Jennings, "On Agent-Based Software Engineering," *Artificial Intelligence*, vol. 177, no. 2, 2000, pp. 277-296.
[6] Protecting America's Freedom in the Information Age, Markle Task Force on National Security in the Information Age. October 2002.
[7] The Big Issue, Jeffrey Pollock. Enterprise Application Integration Journal, April 2002.
[8] Smart Data for Smart Business: 10 Ways Semantic Computing will Transform IT, Michael Daconta. Enterprise Architect, Winter 2003.