zapthink
white paper

Solving Information Integration Challenges in a Service-Oriented Enterprise

SOLVING INFORMATION INTEGRATION CHALLENGES IN A SERVICE-ORIENTED ENTERPRISE

November 2003

Analyst: Ronald Schmelzer

Abstract

As enterprises grow and evolve, they create and store their assets in a wide array of disparate systems and sources ranging from mainframes to relational databases, file systems to directories. However, in order for companies to realize the value of the information stored in these systems, they must integrate and connect the disparate silos of information in the enterprise. As such, today's enterprises face an immediate challenge of connecting relevant systems in a manner that is flexible, cost effective, manageable, and reliable.

Web Services and Service-Oriented Architectures (SOAs) offer compelling solutions for solving integration challenges in a standards-based, loosely coupled, business-oriented manner. However, Web Services address merely the interfaces between systems, applications, and data sources, and offers little, on its own, to solve information integration challenges. As a result, many users that have integration projects require access to data sources that are disparate and heterogeneous. Companies thus require a framework by which they can solve their application and data integration problems not just in a one project, piece-meal fashion, but with a reusable set of components that can be leveraged in future integration projects.

The most advanced Enterprise Information Integration (EII) solutions, in combination with Service-Oriented Integration (SOI), addresses these challenges by providing consolidated access to information and simplified application development through the isolation of the complexities of accessing data from multiple sources. While SOI addresses the integration of application, systems, and processes, bi-directional metadata and model-based EII approaches provide the underlying information integration that allows for coherent access to disparate information.

I. The Challenge of Information Integration

Information is the lifeblood of today's businesses. Companies of any size rely on information, the applications that interact with that information, and the processes and individuals that interact with those applications in order to accomplish any task of importance. As enterprises grow and evolve, they create and store their assets in a wide array of disparate systems and data stores ranging from mainframes to relational databases, file systems to directories. However, in order for companies to realize the value of information stored in these systems, they must integrate the disparate silos of information in the enterprise. And, in order to integrate these disparate silos of information, they must first understand the relationships between their data sources. Once companies have a firm grasp on the information they have, they will be able to manage and access their heterogeneous, disparate information as if it were a single, coherent information source.

As the sources of information grow in the enterprise, leveraging information in an efficient manner becomes increasingly difficult.

As the sources of information grow in the enterprise, leveraging information in an efficient manner becomes increasingly difficult. Furthermore, as the ecosystem of information and applications grow, no single person in the enterprise has a complete view of all the data necessary to perform their day-to-day or strategic tasks. Not only is the information hard to locate, it is often stored in a wide variety of formats that are difficult to quickly interpret. Finally, much of the critical information in an enterprise is often redundantly located in multiple systems in order to guarantee that users can access the information they need reliably and in a format they can process. This redundancy results in unnecessary overhead, management, and data consistency challenges. As such, today's enterprises face an immediate challenge of connecting relevant systems in a manner that is flexible, cost effective, manageable, and reliable.

Successfully surmounting these information and application integration challenges offers a wide range of benefits to companies:

- *Enable the "extended enterprise"* – Integration allows business units and third parties to interact as a connected entity, facilitating functions vital to the flow of commerce. Furthermore, companies seek integration as a means to include new users outside the corporate firewall as participants in their corporate systems and processes.
- *Get a better understanding of customers and business operations* – Integration allows a company to serve its customers and stakeholders better by gaining better access to corporate information.
- *Increase utilization of existing systems and reduce complexity* – Integration allows enterprises to increase utilization of applications that are already deployed in the enterprise, reduce IT infrastructure complexity, simplify management, and reduce the cost of change.
- *Allow systems to evolve* – Provide flexibility in the IT environment by allowing legacy systems to expand to new areas of functionality or provide a way to migrate functionality between systems without wreaking havoc on the rest of the corporate computing ecosystem.
- *Value-add existing applications* – Integration allows companies to make better use of existing systems by continuing to find ways to leverage these systems for new applications.

However, a more strategic driver for organizations is the need to achieve greater *business agility.* Business agility is more than simply being able to respond quickly to change; it also means the ability to leverage change for competitive advantage. For businesses to be agile, their information must be agile as well. Yet, many integration approaches fail to meet this basic business goal.

Most of the issues facing IT fall into three broad categories: complexity, inflexibility, and brittleness.

The plethora of technology present in the IT ecosystem, while intended to address business issues, often presents issues that the IT executive must resolve. Most of these issues fall into three broad categories: *complexity*, *inflexibility*, and *brittleness.* Virtually every enterprise has existing business applications that are difficult to upgrade, difficult to interoperate with, and worst of all, impractical to replace. Furthermore, dissimilar systems tend to be difficult to integrate, each exposing different interfaces with different rules. Integration is therefore an expensive, difficult process that yields inflexible distributed systems. The flipside of inflexibility is brittleness: the risk of failure and other problems inherent in inflexible systems. Traditional approaches to building IT environments lead to a “spaghetti” approach to integration. As a result, when business processes or requirements change, IT departments must either undertake expensive, risky upgrade projects, or simply make do with technology that no longer meets the needs of business. It is the risk inherent in making changes to inflexible distributed systems that is of most concern for enterprises, because of the complex interdependencies among individual systems.

The integration challenge consists of a few major components. Firstly, companies must integrate their internal system in order to provide a cohesive view of data that can power decision-making, customer interaction, and the delivery of integrated products and services. Internal integration is a perennial challenge for most companies since most of these enterprises consist of a mish-mash of heterogeneous systems and architectures that span decades of legacy systems, multiple hardware platforms, operating system versions, database storage technologies, network protocols, component object models, middleware platforms, programming languages, and file formats. Matched with the complexity of internal integration is the desire to connect supply chain partners, affiliate networks, distribution channels, and customers. Cutting across external and internal integration needs, companies need to integrate data and information from a wide variety of data sources. These data sources might be structured, as in the case of databases and enterprise applications, or might be semi- or unstructured such as web pages, PDF documents, Office files, email, media content, or a wide variety of data feeds and formats. The need to access information of so many disparate types from so many disparate sources and locations forms the integration challenge that most companies must deal with today.

The need to access disparate information of many types and sources forms the integration challenge that most companies must deal with today.

The Challenge with Traditional Solutions to Integration

Over the years, companies have pursued a wide variety of integration approaches aimed at solving the underlying requirement for connecting disparate systems in the enterprise. Why are the problems with integration still troubling companies, even though these solutions have been around for a generation or more? On one level, the cause is the lack of standard ways of programming different systems to communicate. For any two different systems, the traditional approach to integration is to write programming code for each system that teaches it how to talk to the other system. Such an approach is expensive and time consuming, and doesn’t scale well or respond to change in a flexible way. This approach to integration is also *tightly coupled*, which means that one programming team must control the integration code on both systems to get them to communicate with each other.

In addition, many of today's organizations look to solve their integration challenges using "point-to-point" integration approaches where systems that need to communicate are connected directly to each other. In this scheme, the number of integration or interconnection pathways that must be established grows geometrically with the number of systems to be integrated. The end result is a tangled web of point-to-point integrations that fails to meet business requirements over time. With traditional integration approaches, the costs for both maintaining and changing systems can become exorbitant, since developers must recode all applications that are impacted by changes. Enterprises are thus faced with an integration problem that grows at a rapidly increasing rate.

With point-to-point integration approaches, enterprises are faced with an integration problem that grows at an exponential rate.

Extract-Transform-Load (ETL)

Historically, companies have sought to solve information integration through the use of *extract, transform and load* (ETL) approaches. The ETL process extracts data from a system on a scheduled basis by copying batch feeds of data from one data source, transforming the data, and then loading the transformed data onto a separate system. However, ETL processes by definition are point-to-point, tightly coupled, and asynchronous since the data load process can occur at any time – most likely in batch processes when the data are already stale. In addition, ETL systems are brittle and hard to change. Once a mapping from a source data to destination format has been defined, any changes to the source file will invalidate the mapping and cause the integration process to fail. Furthermore, managing the mappings and conversion steps is often a complex, cumbersome, and error-prone process. Finally, when integrating with data through ETL means, users frequently bypass the business logic inherent in the system, making it difficult to get timely data that is relevant and related to the business need.

Integration Middleware: EAI and B2Bi

Improving upon the basic integration methods was the movement to implement proprietary middleware integration tiers such as those provided by traditional Enterprise Application Integration (EAI) or Business-to-Business Integration (B2Bi) solutions. These solutions are built primarily on proprietary or system-specific messaging platforms that aim to provide a platform for integrating and communicating with various business components. The typical method for accessing these systems is through a wide assortment of pre-built adapters that provide bi-directional connectivity to many types of applications and data sources. In simple terms, the way these integration solutions work is by extracting or inserting data from these various adapter-enabled systems, transforming the data and converting their representation or schema to a different format, and then shipping the data to their destination.

However, experience has shown that EAI and B2Bi solutions have been far from optimal ways of implementing application and data integration solutions, First, the centralized nature of many EAI implementations (especially "hub-and-spoke" architectures) are not particularly efficient or scalable, and is often the root cause for many failed integration solutions. In addition, the up-front costs for EAI solutions are much higher than other integration approaches. In a typical EAI solution, end-users must spend from tens of thousands to millions of dollars on software licenses and server systems prior to completing any integration. The actual integration project itself typically ends up costing many times more than the initial costs and can easily dwarf the costs of custom integration projects. Furthermore, when systems, business processes, or major assumptions change, EAI system costs can spike. In fact, we say that EAI systems "pour concrete on business processes," since they tend to solidify existing processes rather than enable an IT environment that allows companies to deal easily with change.

In addition, EAI solutions don't work well across more than one company or corporate division, because there is no single programming team that can control all the interfaces. And, for systems that are not accessible on the network, EAI-type solutions fail to provide any solution. EAI fundamentally is a centralized integration process – which doesn't work well for decentralized data. Most importantly, however, EAI and B2Bi solutions are tightly coupled. They require developers to make assumptions of the particular IT ecosystem, and then rigidly implement to those assumptions. Traditional EAI and B2Bi solutions are thus brittle, inflexible, and incapable of handling the agile needs of today's businesses.

Enterprise Information Integration (EII) improves upon traditional integration approaches by making multiple, different data sources appear as a single, comprehensive data store.

Advanced Enterprise Information Integration – Steps in the Right Direction

Enterprise Information Integration (EII) improves upon traditional integration approaches by focusing not on the application interfaces, but rather on the data that is exchanged between systems. EII systems aggregate data and coordinate transactions across enterprise data sources. The goal of an EII implementation is to provide a cohesive view of enterprise data, in effect making multiple, different data sources appear as a single, comprehensive data store. Through this approach, users not only gain consolidated access to information, but also simplified application development through isolation of the complexities of accessing data from multiple sources.

It is important to note that the term "EII" has enough generality that not all vendors who claim EII solutions actually meet the criteria stated above. As a result, what we are really talking about here are "advanced" EII solutions, which are differentiated from other integration approaches in its use of metadata as a means of assembling a coherent view of, and bi-directionally interacting with, disparate information. Simply put, metadata is the information about the various data sources that provides context and understanding in how to process the information. For example, metadata provides knowledge on what kind of information is in a particular data source, how it is named, formatted, indexed, organized, and the context in which to process it. In this way, metadata provides the user a common and integrated view of an organization's information infrastructure. Therefore, the key to making advanced EII work is not just facilitating how pieces of information relate to one another and providing a means of accessing that data, but also managing the many types of metadata surrounding that information.

The Requirements for Information Agility

While advanced EII has been a positive step for most companies looking to solve their information integration needs, what companies need to solve their integration cost and complexity issues is a *loosely coupled* approach to integration—one that scales and does not require control of systems on both ends nor an intimate connection between the requester of information and the provider. However, for loose coupling to be a reality, there must be standard, established ways of handling integration so that any company that follows the standards can be confident that their systems will interoperate with other systems that the company wishes to communicate with. XML represents the standards-based way of representing information, and *Web Services* have come to signify the overall movement toward standards-based approaches to distributed computing. This standards-based approach to loosely coupled integration is what the promise of XML and Web Services are all about.

To support loose coupling at the Data level, what is needed is a real-time, accurate, integrated, and efficient method for aggregating information from multiple sources.

To support loose coupling at the Data level, what is needed is a real-time, accurate, integrated, and efficient method for aggregating information from multiple sources, bringing together both structured and unstructured data, and representing that package of information in XML formats to feed operational data requirements of Web Services, Service Oriented Architectures (SOA) and service-oriented processes. This method needs to connect to many different back-office financial systems as well as customer-facing front-office and B2B systems.

Rather than thinking about how to get information into or out of different systems, users must think about how to expose a system in a Service-oriented manner to whatever system cares to access it. In this way, corporate IT is released from thinking of information in a point-to-point fashion, but rather as freely available as a network of Services. In this way, the organization can reuse existing functionality in different ways – for building new, composite applications, or extending existing functionality. As a result, this new class of integration solution must be:

- *Loosely-coupled* – Consumers don’t need to have knowledge beforehand about a given piece of system functionality, other than where to find it. Application functionality and the programs that invoke them can be changed independently of each other, instead of requiring a redesign of the involved components.

- *Coarse-grained* – Rather than interacting with a large set of detailed, fine-grained APIs, users can interact with systems through *coarse-grained,* business-level interfaces that roll up the functions of many different API calls into a small number of business-oriented messages. Although low-level, fine-grained, services already exist in the enterprise, they exist separately from each other in discrete silos, and usually are not participatory in a service-oriented architecture. Often, the first step is simply to create fine-grained, low-level services and then bring them together as higher-level Services to produce a discrete set of capabilities that didn’t exist before.

- *Standards-based* – Instead of utilizing proprietary or closed APIs that require users to learn the intricacies of a particular vendor’s platform, integration tools that successfully address the challenges of business agility will be standards-based, lowering the cost of integration and allowing the widest possible range of developers.

- *Process-driven* – Instead of dealing with concrete requirements from business, users must be able to respond to changing requirements. The entire architecture—from the hardware on up—must reflect the business agility requirement, because any bottleneck in an integration implementation can substantially reduce the flexibility of the entire IT environment, and hence the business as well. Products and methodologies that allow line-of-business users to create and manage processes facilitate business agility.

In essence, companies should seek to combine the best that an EII system offers for metadata-based information integration with that of standards-based, loosely coupled Service-Oriented Integration. In this way, companies can aim to solve their long-standing integration issues.

Companies should seek to combine the best that an EII system offers with that of standards-based, loosely coupled Service-Oriented Integration.

II. Service-Oriented Integration

Into this tough environment an evolutionary, standards-based approach to architecting IT application functionality is gaining traction at enterprises across many industries. Known as *Service-oriented architecture* (SOA), this new way of thinking about how to access application functionality and integrate IT resources in the enterprise promises to address all of the integration issues listed above.

SOAs are an approach to enterprise software business systems and business applications that considers these resources as services available and discoverable on a network. Such services provide functionality to the business while hiding the underlying implementation details. Providers of these services must be able to publish information about them in a *service registry*, where service consumers, or requestors, can then look up the services they need and retrieve the information they need about those services to find and use, or “bind” to them.

Service-Oriented Integration

The vision of using Web Services, or any Service-Oriented Architecture, for integration is known simply as Service-Oriented Integration (SOI). Rather than explicitly declaring how systems will interact through low-level protocols and object-oriented architectures, as was the case with previous EAI and B2Bi efforts, SOI provides an abstracted interface with which systems can interact. Systems merely need to expose their capabilities as Services, and other systems that choose to interact with them can simply discover those services and bind to them either at runtime or design-time. Of course, Web Services aren’t the solution to these integration problems; they just provide a technology on which to produce solutions.

Rather than thinking about how to interconnect systems, we can think about merely how to expose a system as a set of loosely-coupled, coarse-grained Services.

Most existing integration solutions involve the connecting of different applications and data stores to each other in a point-to-point approach or hub-and-spoke approach. The Service-oriented vision changes that. Rather than thinking about how to get information into or out of different systems, we can think about merely how to expose a system in a Service-oriented manner to whatever system cares to access it. In this way, we release ourselves from thinking of information in a point-to-point fashion and instead think of information as freely available on a bus or web of Services. In addition, SOI advocates keeping all integrations loosely coupled. The point of integration is to allow arbitrary applications, systems, and data stores to communicate without concern as to the other system’s requirements, and SOI enables that reality. From a Services point of view, the exposure of information, application functionality, and processes as dynamically bindable, discoverable objects can be considered ad-hoc integration, vs. tightly coupled, fine-grained integration.

A comparison of the different integration approaches can be seen below:

Table 1: Comparison of Integration Approaches

Integration Approach	Advantages	Disadvantages
Custom integration (developer coding)	➢ Simple to implement ➢ Can use existing tools ➢ Short time-to-market	➢ Doesn't scale ➢ Management nightmare ➢ Difficult to add systems ➢ Very tightly coupled
Data-level integration through direct access to data sources	➢ Easy to implement ➢ Little change to existing application logic ➢ Use some existing tools	➢ Does not allow addition of new functionality ➢ No addition of new business logic ➢ Hard to integrate multiple data streams ➢ Tightly coupled
EAI / B2Bi Integration	➢ Asynchronous model ➢ Integrate business logic ➢ Rules-based transformation ➢ Hub-and-spoke or bus model	➢ High TCO ➢ Proprietary ➢ Tightly coupled ➢ Problem in synchronous models ➢ Scalability ➢ Complex to manage
Service-Oriented Integration	➢ Standards-based ➢ Loosely coupled ➢ Asynchronous and synchronous ➢ Easy to add new logic ➢ Lower cost than EAI ➢ Business agility ➢ Multiple models ➢ Bi-directional transaction support[1]	➢ Standards immature ➢ Missing robust security, reliability ➢ Needs new tool sets

Source: ZapThink, LLC

Benefits of Service-Oriented Integration

There are a number of major differences between tackling integration from a Service-oriented point of view and the traditional approach to EAI taken by existing vendors. Many of these differences represent real improvements over how integration was performed prior to SOI solutions. In particular, some of the key benefits of implementing SOI technology for integration include:

- Reducing the cost and complexity of managing IT infrastructures
- Providing a uniform platform for B2B exchanges and eMarketplaces
- Moving away from proprietary technologies and solutions
- Enabling application and data reuse
- Simplifying business modeling
- Providing right-grained (either fine-grained or coarse-grained as needed) access to data, functionality, and logic

One major benefit that SOAs provide is that they facilitate the operation of heterogeneous IT environments. Instead of "rip and replace," moving to an SOA means building bridges among different systems and applications, rather than

[1] Bi-directional transaction support is not a feature of most EII offerings, however advanced systems such as that of MetaMatrix do support XA compliant transactions.

throwing them out. In fact, the business value of SOAs is so dramatic that enterprises across the world are considering how to transition their existing IT infrastructures to SOAs.

SOI simplifies system integration by providing a single, simple architectural framework in which to build, deploy, and manage application functionality. Rather than building increasingly larger sets of application functionality, users can build small, finely grained application code that can be published, dynamically located, and bound to without a significantly complicated object-oriented brokering or messaging solution. With simplified components comes reduced cost. Web Services allow users to build complex systems out of very simple components without sacrificing power. This is the key vision that the Service-Oriented Architecture (SOA) brings to corporate IT.

SOI simplifies system integration by providing a single, simple architectural framework in which to build, deploy, and manage application functionality.

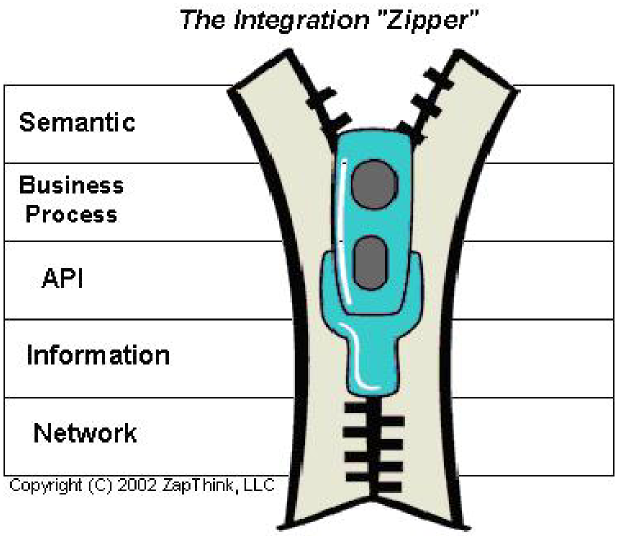
The proliferation of tools and the fact that SOI solutions are based on open standards means that SOI solutions can be simpler to design, implement, and maintain. Rather than having to learn the intricacies of different systems, communications technologies, and platforms, users can implement a more widely understood SOI methodology for integration.

As a result of their simplicity and reliance on open standards, Web Services-based SOI solutions can be less expensive to implement than traditional EAI and B2Bi solutions. Clearly, one of the motivations towards moving to a Service-oriented approach to integration is that its simplicity and use of standards will reduce initial and long-term cost of integration. The number of pre-packaged solutions available to developers of Web Services will no doubt increase dramatically, as will the number of skilled developers. Part of this reduction in cost is that Web Services, upon which SOI is based, can be implemented using relatively simple technologies and languages, including scripting languages.

How Advanced EII adds to the SOI Value Proposition

ZapThink sees that integration is not a binary phenomenon, but rather consists of a spectrum of different levels to which systems are integrated:

Figure 2: The Integration “Zipper“



Integration problems never go away. Rather, as standardization and commoditization of technologies solve the problems on the lower levels of the integration stack, higher-level integration issues remain. These issues connect less and less to specific implementation decisions; instead, they are increasingly business and concept-oriented. Instead of having to worry about the connectors, adapters, and interfaces between systems, businesses will need to worry about how to map the way that they represent information itself. Even if the connection between businesses and systems were seamless, companies must still resolve the issue of the meaning of the information passing between the organizations.

What makes semantic integration a challenge is two-fold: first, the representation of information and the information itself are often bound tightly together; and second, that information frequently lacks context. Developers often think not of the data itself but rather the structure of those data: schemas, data types, relational database constructs, file formats, and so forth--structures that don't pertain directly to the information at hand, but rather our assumption of what the data should look like. In tightly-coupled architectures, data structures are absolutely necessary, since they provide systems a way of coping with the information they are being fed.

However, in a standards-based, loosely coupled architecture, when the barriers to application integration are removed, instead of being helpful constructs, these various data structure representations actually get in the way. How information is stored and represented interferes with the meaning of that information. To be more precise, the meaning of information and the structure of that information aren't one and the same. For example, "August 7, 2003" is a date for sure, but whether or not it is stored as a string, date type, or integer shouldn't matter. Yet, developers often needlessly combine the structure and meaning together inextricably.

In order for data to flow unimpeded in a Service-Oriented Architecture, Service providers must isolate requesters from the underlying data structure assumptions.

Thus, when one developer's assumption of a particular structure for some datum conflicts with another's representation, you get an impedance mismatch - in other words, a data integration problem. In order for data to flow unimpeded in a Service-Oriented Architecture, Service providers must isolate requesters from the underlying data structure assumptions. The issue here is therefore one of loose coupling. While we might loosely couple application interfaces through the use of SOAs, if we deal with data the same way we've always done - by imposing the data structures of Service providers on Service requesters, the result is every bit as tightly coupled as previous architectural approaches. In order to provide the promise of seamless data integration, we must transcend simply loosely coupling the application interface and in addition provide loose coupling at the semantic level.

Yet, this is primarily the problem that advanced EII approaches have been charted with solving. As such, there is a compelling value proposition in combining the loosely coupled, standards-based, coarse-grained integration approaches of SOI with the information-centric integration capabilities of EII. There are several key ways in which advanced EII solutions provide value in addition to the value inherent inSOI solutions:

- *Information abstraction layer* –A key requirement of information integration is that users need to be insulated from the underlying complexity of these sources. As a result, EII solutions can manage information access at a logical level that can be described in business terms, rather than the physical access constraints of a particular data source. In addition, EII provides information modeling tools that separate the logical, business-oriented information consumption views from the back-end, physical data sources. Advanced EII solutions provide this abstraction layer not only for query purposes, but for bi-directional access as well.

- *Data discovery* – Even when applications and processes can be easily identified, the source of the data and information to power those systems are not as easily locatable or even identifiable. EII systems enable users to create an inventory of available data sources, in order to determine the information and metadata to be integrated.
- *Metadata management* – Advanced EII systems provide further information visibility by providing management of the metadata surrounding those data sources. This capability enables users to determine if information being exchanged is redundant, overlapping, or complementary, as well as provides information on how an information source relates to another.
- *Data Manipulation* – EII solutions realize that information is not typically consumable as-is by systems, and as such supports mapping between formats, transformations, and application of business logic to data. In addition, EII facilitates data exchange by smoothing over the differences between synchronous and asynchronous access to data sources.
- *Bi-directional Data Integration* – In addition to providing simple data access, EII solutions also enable input of data into multiple data sources from a single feed. This capability eliminates the need for replication and synchronization that is the bane of most company's integration problems.

So, it's clear that advanced EII offers promise for furthering the SOI value proposition, but how can users actually leverage these EII solutions in a way that is closely aligned with the goals of SOI?

III. Leveraging the Metadata Repository for Service-Oriented Integration

Management of metadata is the key to information integration. Metadata, or information about data, consists of information that helps systems understand the context and/or environment of a piece of information. For example, metadata could include data schemas, descriptive text, data access requirements, security constraints, design documents, or other structural or informational context for the underlying data. Companies thus can leverage metadata to build a complete picture of the organization's data sources.

Management of metadata is the key to information integration.

However, since there is just as much information in the metadata as there is in the data itself, companies require a data store, or repository, that is tasked with storing, managing, and facilitating distributed retrieval of metadata. Metadata management involves storage and access to metadata information in a metadata repository, as well as a means to define and manipulate metadata models through a graphical interface. Companies can then leverage the repository to enable easy-to-navigate views of metadata, perform data modeling and analysis, discover data assets, and simplify cross-data source information searches.

Information Models: Physical, Logical, and Virtual

In order to make an EII solution work in the context of loosely-coupled Service-Oriented Integration, the source of the data must be abstracted from the consumers of the data – a concept known as *data abstraction*. The best way to understand abstraction from a data perspective is to consider three different models for information in the enterprise:

- *Physical model* –The physical model represents the actual sources of data and the specific, tightly-coupled representations of the information contianed in those sources. For example, a physical model might represent a collection of relational databases, file systems, and directories, and the schema necessary to interact directly with those data sources.
- *Logical model* – Represents the business processes, system-to-system interaction, and aggregated information that is used to enable business transactions or decision making. The logical model doesn't identify the particular data sources or systems, but rather shows how the information is to be used.
- *Virtual model* – A virtual model is an instantiation of a logical and physical model that contain a particular organization's or end user's view of the information. These models describe not only what information the end user needs (an instance of a logical model), but maps the information to the physical data sources from which the information ultimately comes. This view describes the available information as data-consuming users and applications want to see it, in terms of a single virtual data source, and provides a mapping or transformation to show how virtual metadata is derived from the physical metadata.

In order to get any value from a service-oriented EII solution, an organization must model its physical data sources, logical information flows and processes, and virtual data views.

In order to get any value from a service-oriented EII solution, an organization must model its physical data sources, logical information flows and processes, and virtual data views. Now, it's important to note that the modeling phase doesn't have to be done in one, large initiative all at one time. Rather, a best practice for modeling activities is to approach it incrementally – companies should model on a department-by-department, or IT project-by-project basis, thus accomplishing the goals of modeling organically. As additional integration initiatives are implemented, the metadata management capabilities can be leveraged to enable the re-use of physical, logical, and virtual models. Once the enterprise has accomplished these tasks, the organization will have a coherent and complete map to the organization's available data and current uses. These models can then serve as the underpinnings for information integration in the context of Service-Oriented Integration solutions.

Finally, once the models are registered in a repository, users can dynamically discover available data sources, the means to access those data sources, and exploit a unified query mechanism for access to the information. Users can access the repository either at design-time, as part of tightly-coupled application development, or at run-time as part of a loosely coupled, coarse-grained SOI solutions. In this manner, while SOI addresses the integration of application, systems, and processes, the metadata and model-based EII approach provides the underlying information integration that allows for coherent access to disparate information.

IV. The MetaMatrix Solution

Since their founding in 1998, MetaMatrix has been a pioneer of Enterprise Information Integration (EII) solutions. The company advanced the state of the art of information integration with one of their two core products, MetaBase, that enables organizations to discover and map the disparate data sources in their IT environment. Organizations can use MetaBase to create physical models that describe how their data sources store information as well as the virtual metadata models that describe how the organization's users and applications use that information.

MetaBase consists of two major components: MetaBase Repository and Metabase Modeler. The MetaBase Repository offers a single storage location for metadata information about all of the organization's data assets. The Repository contains version control and check-out/check-in model management capabilities, as well as a robust search mechanism to locate information structures based on descriptions, names, and keywords. The MetaBase Modeler lets organizations import the metadata information from a variety of sources, including JDBC-compliant relational database management systems, XML Schema, or XML Metadata Interchange (XMI) metadata models. Models imported and created in MetaBase Modeler can also be updated in real-time using wizards and other automatically updating tools. Users can also create metadata by hand using a hierarchical tree, a UML diagram, and tables.

By connecting information sources to the MetaMatrix Server, enterprises gain real-time access to information, single sign-on security, and a single access point to disparate information sources.

Leveraging MetaBase is the company's flagship product, MetaMatrix Server. The MetaMatrix Server provides an EII platform that allows companies to integrate their enterprise information systems into a "virtual database" that provides coherent access to disparate data sources. By connecting information sources to the MetaMatrix Server, enterprises gain real-time access to the information, single sign-on security, and a single access point to the disparity of information sources. The MetaMatrix Server product uses the metadata models generated in the MetaBase Modeler to resolve queries posed by user applications. The MetaMatrix Server parses queries based upon the metadata information and distributes the queries to the end data sources, aggregating the results, or inserting data into data source end points.

MetaMatrix Value Proposition

MetaMatrix provides a number of compelling value propositions that combine the best features of Enterprise Information Integration with the benefits of Service-Oriented Integration. In particular, MetaMatrix provides:

- *Loosely coupled integration through a data abstraction layer* – MetaMatrix enables loosely-coupled data integration through the use of a data abstraction layer that isolates information consumers from the specifics of the data sources. MetaMatrix provides a logical layer between the disparate data sources and application developer or application. The MetaMatrix Server uses the logical layer to integrate information so that it looks like a single database to the developer or the application. Inside this layer, MetaMatrix metadata mappings point to the sources of each element. By using a virtual database as the data source, applications do not connect to any physical sources directly. This allows the physical data sources to change without affecting downstream applications.
- *Reduced cost of integration through standards-based information integration* – MetaMatrix uses a wide range of standard-based communication protocols such as JDBC, ODBC, and SOAP, and standards-based data representation languages such as XML and SQL. MetaBase also takes advantage of several industry standards for metamodel management to ensure interoperability and future-proofing of the metadata repository, such as the Object Management Group's (OMG) Meta-Object Facility (MOF) as a means to provide a common description and relationships for metadata of disparate information

sources, XML Metadata Interchange (XMI) for model sharing and archiving, and Common Warehouse Metamodel (CWM) for model collaboration.

- *Integration reliability through federated data management and access* – MetaMatrix Server provides federated data management through the use of distributed and/or redundant query servers that provide unified access to enterprise information.
- *XA Compliant Transactions* -- Transactions are a logical unit of work that either modifies some state, performs a set of operations, or both. An individual transaction may involve multiple data and logical operations, but these operations always occur as an indivisible atomic unit, or they do not occur at all. The MetaMatrix System has the capability to Create, Read, Update, and Delete data entities. This enables customers to treat a MetaMatrix Server spanning multiple physical data sources as if it were a real database, with not only read capability but also write-back capability.

By connecting data sources in a loosely coupled manner using MetaMatrix Server and MetaBase, enterprises can finally realize the benefits of business agility through real-time access to information, and central, cohesive access to disparate information sources.

V. Conclusions

Integrated access to aggregated information has been a headache and challenge for most enterprises for decades. Traditional data and application integration solutions have offered some relief, but at great expense, complexity, and rigidity. Organizations clearly need a solution that not only can integrate a wide variety of data in the enterprise, but also embrace their heterogeneity – and do so without sacrificing the critical performance businesses require. While traditional EAI and B2Bi solutions have offered some relief, those solutions have offered limited ROI at great expense, complexity, and brittleness. Adopting standards-based Service-Oriented Architectures for integration, specifically Web Services-based Service-Oriented Integration (SOI) techniques provide a loosely coupled, coarse-grained approach to connecting systems that promises to significantly improve the economics of integration and continued realization of IT value.

Adopting standards-based SOAs in combination with EII techniques provide an approach to integration that promises to significantly improve the economics of integration and realization of IT value.

Applying SOA to solve integration of application functionality doesn't make all the problems of integration go away - it simply brings the issue of semantic integration to the surface. Companies should be thinking of semantic integration in the context of SOA - using the techniques of encapsulation and loose coupling to make data flow seamlessly. The application of Enterprise Information Integration (EII) solutions, such as those offered by MetaMatrix, to SOI approaches provide compelling solutions to challenging semantic integration problems.

Copyright, Trademark Notice, and Statement of Opinion

The information contained herein has been obtained from sources believed to be reliable. ZapThink disclaims all warranties as to the accuracy, completeness or adequacy of such information. ZapThink shall have no liability for errors, omissions or inadequacies in the information contained herein or for interpretations thereof. The reader assumes sole responsibility for the selection of these materials to achieve its intended results. The opinions expressed herein are subject to change without notice. All trademarks, service marks, and trade names are trademarked by their respective owners and ZapThink makes no claims to these names.

About ZapThink, LLC

ZapThink is an IT market intelligence firm that provides trusted advice and critical insight into XML, Web Services, and Service Orientation. We provide our target audience of IT vendors, service providers and end-users a clear roadmap for standards-based, loosely coupled distributed computing – a vision of IT meeting the needs of the agile business.

ZapThink's role is to help companies understand these IT products and services in the context of SOAs and the vision of Service Orientation. ZapThink provides market intelligence to IT vendors who offer XML and Web Services-based products to help them understand their competitive landscape and how to communicate their value proposition to their customers within the context of Service Orientation, and lay out their product roadmaps for the coming wave of Service Orientation. ZapThink also provides implementation intelligence to IT users who are seeking guidance and clarity into how to assemble the available products and services into a coherent roadmap to Service Orientation. Finally, ZapThink provides demand intelligence to IT vendors and service providers who must understand the needs of IT users as they follow the roadmap to Service Orientation.

ZapThink's senior analysts are widely regarded as the "go to analysts" for XML, Web Services, and SOAs by vendors, end-users, and the press. They are in great demand as speakers, and have presented at conferences and industry events around the world. They are among the most quoted industry analysts in the IT industry.

ZapThink was founded in October 2000 and is headquartered in Waltham, Massachusetts. Its customers include Global 1000 firms, public sector organizations around the world, and many emerging businesses. ZapThink Analysts have years of experience in IT as well as research and analysis. Its analysts have previously been with such firms as IDC and ChannelWave, and have sat on the working group committees for standards bodies such as RosettaNet, UDDI, CPExchange, ebXML, EIDX, and CompTIA.

Call, email, or visit the ZapThink Web site to learn more about how ZapThink can help you to better understand how XML and Web Services impact your business or organization.

ZAPTHINK CONTACT:
ZapThink, LLC
11 Willow Street, Suite 200
Waltham, MA 02453
Phone: +1 (781) 207 0203
Fax: +1 (786) 524 3186
info@zapthink.com

ZapThink, LLC • 11 Willow Street, Suite 200 • Waltham, MA 02453 • info@zapthink.com • www.zapthink.com