Service-Oriented Architectures XML

The Case for Enterprise Data Services in SOA

02/28/2008

By Jeff Pollock, Senior Director, Fusion Middleware, Oracle

Overview

With the hype surrounding Service-Oriented Architecture, people often forget that SOA is mostly about plumbing, and that a substantial portion of the top-line business value of SOA is derived directly from the data it pushes around.

Today, the data universe of Business Intelligence (BI), Data Warehousing and Master Data Management (MDM) still lives largely outside the scope of SOA. But an enterprise view of Data Services may well bring these worlds together.

When properly architected and executed, Data Services can provide the link that unifies conventional data systems with the emerging SOA paradigm. By offering a decoupled data façade and an easily virtualized API, Data Services can give SOA the opportunity to establish system control.

Part 1: The Primacy of Data

It's always been about the data. Decades of punditry about EAI, ETL, MDM and SOA lead us to the same conclusions -- data is king in enterprise software.

Sometimes the enterprise software sector loses sight of that simple reality. In the past 15 years, with the rise of Java, the hype surrounding EII, EAI and SOA, and the rise of XML, and quietly, the billions spent in ETL projects -- it's too easy to forget why we build and buy all that infrastructure. We do it for the data.

There has been more data created since 2000 than in all of human history preceding then. (Figure 1)

ADVERTISEMENT



Our Popular Webinars

Technology in the New Environment

Best Practices for Business Service Management (BSM)
Hosted by HP and Featuring Peter O'Neill of Forrester Research

SOA in Financial Services Live Panel: Visibility, Control and Evolution: Building on SOA to Meet Today's Financial Services Industry Challenges

Federation and User Centric Identity – The Future Secure Identity Architecture for Your Businesses Partners and Consumers

Banking on SOA Governance: Guiding IT Investment for Business Value

More Webinars

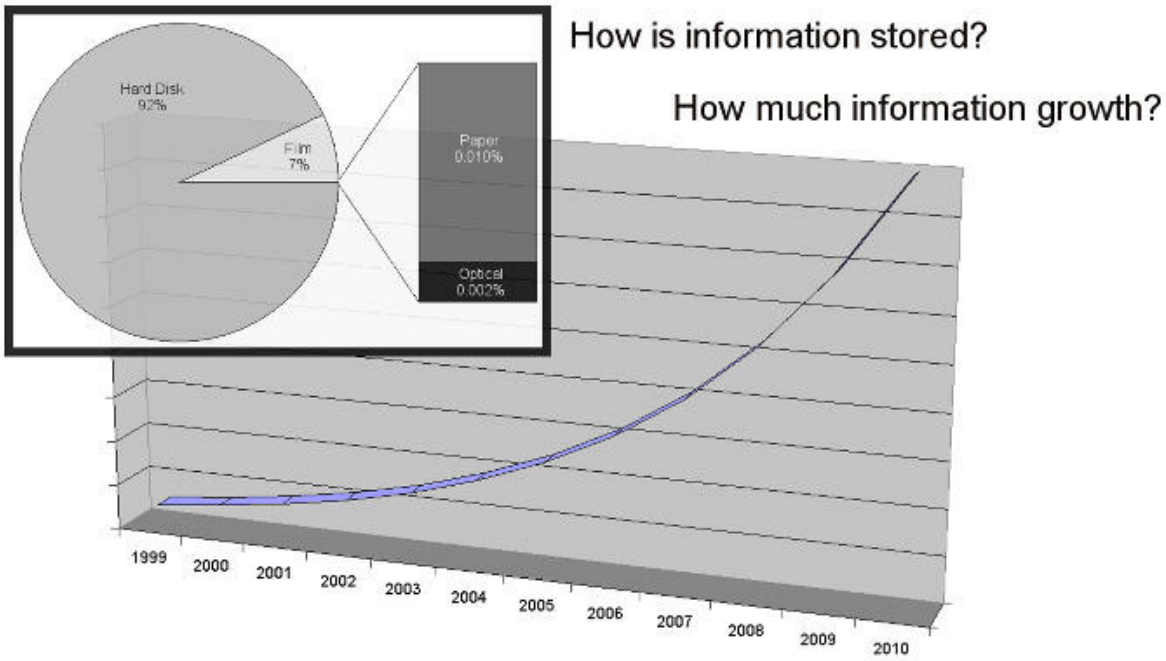


Figure 1: Information Explosion (Adaptive Information, Wiley & Sons 2004)

These trends show us that the data problem is getting worse, not better.

Enterprise infrastructure is surprisingly unchanged since the early 1990s when Message-Queues (MQ), Transaction Processing Systems (TPS), and ETL tools were really the backbone of enterprise software. Guess what? They still are. Despite the growing adoption rates of BPM, SOA, ESB, and EII – the MQ, TPS, and ETL backbones are still there.

The strain of all that new data and the demand for mature tooling has paradoxically made the existing, proven software infrastructure look pretty attractive. Most new XML-centric solutions for data can't scale to the mult-terabyte sized problem that is typical of a Global 2000 business. Thus, a knowledgeable architect will revert to the proven patterns of RDBMS as the backbone of a data architecture using MQ, TPS, and ETL interfaces as the pipes for pushing all that data around.

Why not SOA for data-centric architectures?

Part 2: A Role for SOA in Data-Centric Enterprise Architectures?

When the SOA craze started in 2001, we thought it was magic. Remember the promises of dynamic discovery? Human readable messaging? Simple XML data objects? But soon enough, the problems started: competing vendor specs, security loopholes, performance problems, etc.

In 2008, the good news is that SOA has finally matured into an enterprise class infrastructure. Far from the original hope of solving all integration problems, the main tooling for SOA can finally supplant the long-held dominance of MQ and TPS systems. Both the reliability and performance of grid-based SOA is strong enough for even the most demanding problems.

However, SOA is still not best for ETL, MDM, BI and data integration.

The average data integration use case is beyond SOA's core strengths. An average use case might involve loading a few gigabytes of data from one database to another, applying transformations to change the shape of the data from third-normal-form to a multi-dimensional model. This use case supports line-of-business demands like Reporting, BI, Performance Management, Financial Planning and other analytic capabilities. SOA is inappropriate for these

data use cases because of poor bulk data transformation performance and inefficiency.

Nearly all SOA frameworks operate in a Java container, which is a substantial disadvantage when gigabytes of data need to be consumed into a Java Virtual Machine. Likewise, the SOA paradigm for working with data is XML -- nearly all SOA frameworks require the data to be converted to XML for it to be orchestrated and transformed. But a single gigabyte of data will multiply to five or ten gigabytes of XML data merely because of the additional tags, schema and angle brackets. (Figure 2)



Figure 2: Added Data Bulk With XML Markup

After all, XML is still best as a document and message format. For a while, the SOA buzz had everyone thinking that XML is a data language, but it's not. A simplification of SGML, XML was only ever intended to provide a well-structured, standard way of marking up documents and messages. The core model of XSD is actually called Infoset, a tree-like structure to define which XML items are allowable. But Infoset isn't supposed to be a data model in the same way that regular data models are. For instance, XSD structural semantics are imprecise -- the nesting of a tag can mean different things in different applications and that meaning isn't enforced by standard parsers. This is one reason why pure XML databases are exceedingly rare.

In fact, neither of the early Java/XML definitions of SOA Data Services are truly scalable to enterprise sized problems. The Java definition, largely heralded from a host of standards like JDO, SDO, DAS, DTO, etc is really about (a) trying to define patterns for interfacing Java with relational data and (b) standardizing the APIs for moving those objects or components around between applications and Java containers. Thus, the two main market definitions of Data Services, XML-based and Java-based (Figure 3), have limited applicability for real-world enterprise sized data problems.

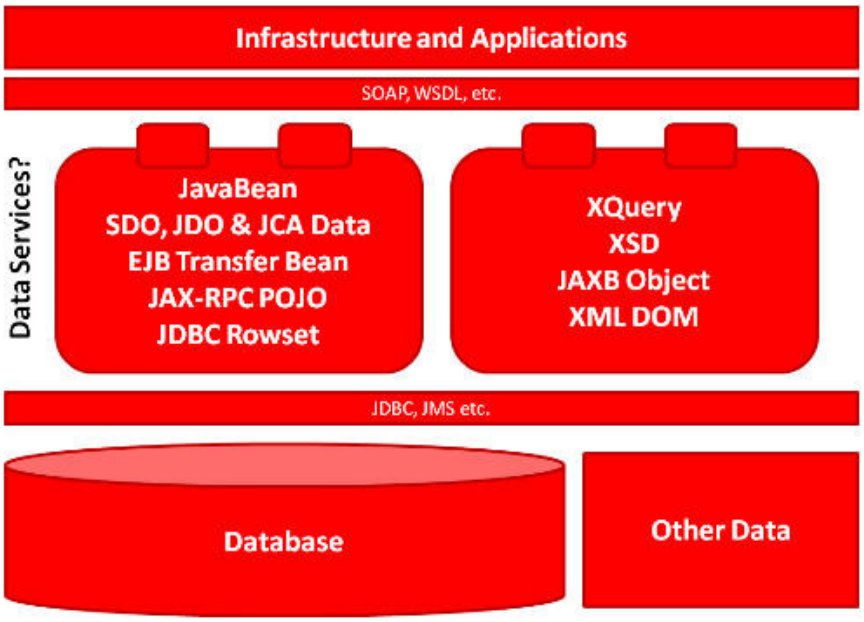


Figure 3: A Conventional Java/XML View of Data Services

The simple and unfortunate reality is that enterprise data requirements are hard, and the dreams of SOA only solution for all enterprise data are likely to remain dreams.

Enterprise data requirements are fundamentally too complex and too closely driven by the high-volume, mutli-dimensional nature of BI systems to entirely be serviced from a messaging layer alone. Further, valuable patterns and lessons about enterprise data services actually precede the invention of SOA, and can exist in harmony or completely independently from the SOA infrastructure itself.

So, given this decoupling of "services for data" from SOA, what does SOA have to do with data services?

Part 3: What Are SOA Data Services?

Despite the inability of SOA to crack into that data management foundation, there is mounting evidence that suggests that harmonizing data and service architectures may yet generate substantial new benefits. These benefits derive from the use of SOA as a control point for data infrastructure, not a replacement of it. Thus, SOA Data Services are decoupled end-points that expose highly optimized components for working on all types of data.

Data services themselves don't need SOA.

Depending on how you may personally define data services, it is quite easy to claim that data services have been an institutionalized part of software infrastructure since the rise of EDI (Electronic Data Interchange) services between financial institutions in the 1960s. Later, key data service patterns became commonplace in the 1980s with the rise of Object-Oriented design principles. Most recently, data services in Java actually pre-date the notion of SOA data services by a few years.

Technically, a data service should exhibit several of the following attributes:

- Contract-based bindings – for design-by-contract, WSDL/SCA for example
- Data encapsulation – access to data via APIs only, indirectly
- Declarative API – some type of query-able API in addition to regular bindings
- Decoupled binding metadata – API descriptors are themselves part of some model
- Decoupled data schema metadata – data schema is separate from API

Perhaps the notion of a data service is more about an ideal. Data services may be about the ideal that there can be a single, shared control point for all important business data. Data services should expose control points for data that are easy to access, publish, and discover. So, in a most basic way, the data service may simply be a stereotype -- a label, or tag -- used to mark a particular software component' s purpose for existing.One myopic view of data services is that they provide only Enterprise Information Integration (EII) style federated queries. Several small vendors have staked a claim that EII by itself supplies data services as federated queries and XQuery or SQL-based data views. But these cache-based delivery mechanisms simply equate to a data hub in practice. In fact, business requirements for true (non-cache-based) query federation are exceedingly rare in actual practice -- EII market share proves this -- and are only a very small aspect of real world data services.

With an eye towards enterprise sized problems, Data Services should encompass several different data delivery patterns and formats. SOA pundits often assume that XML is the only desirable delivery format, but for a data solution to be truly useful for the enterprise, it must support several different delivery styles.

It sounds trite, but the simple advice for Data Services is to always use the right tool for the job. Too many SOA fans see XML as the solution to every problem when in fact there are many tools far better optimized for the non-XML data formats that are pervasive within typical large businesses. SOA is best conceived of as a framework for common control points, process management and re-configuration -- not as a universal data layer.

Thus, the low-hanging fruit for Data Services may be boring Web Services with simple data actions, or thin SOA façades for wrapping conventional data technology. But these starting points are perhaps the most useful and common-sense ways to start a multi-year Data Services plan that truly serves the enterprise.

The Data Services architect must remain acutely aware of the application performance requirements and the

additional latency that SDO and XQuery approaches cause in the Data Services layer. Neither SDO nor XQuery are required to successfully deploy Data Services. In fact, a comprehensive Data Services infrastructure will exhibit a range of architectures, functional services, and delivery styles, including:

- Architecture Patterns for Data Services – where the service executes
  - Basic WSDL/XML Façade – simple WSDL façade to a data source
  - Java SDO Proxy – Java abstraction for diverse data sources
  - XQuery/XML Proxy – XML layer abstraction for diverse data sources
  - SQL/RDB Proxy – SQL layer abstraction for diverse DBMS schema
  - Data Service Façade – a pass-by-reference API for conventional data services (such as: replication, migration, integration, transformation, master data…)

- High Level Functional Data Services – what the service does
  - Master Data Service – lifecycle maintenance of golden records
  - Batch Data Services – optimized bulk movement & transformation
  - Data Access Service – fetching and changing normative business data
  - Data Grid Services – optimized caching and clustering of data objects
  - Data Quality Services – automated cleansing and de-duplication
  - Data Transformation Services – centralized transformation components
  - Data Event Services – monitoring for data state, changes and rules

- Data Distribution Styles for Data Services – how the data is delivered
  - RPC-style Delivery – remote invocation using regular request-reply
  - Event-based Delivery – publish/subscribe via queuing type system
  - Process-based Delivery – transactions via BPEL or other long-lived XA
  - Object Delivery – via marshaled objects in the application language
  - Bulk-style Delivery – low level, direct to/from source persistence layer

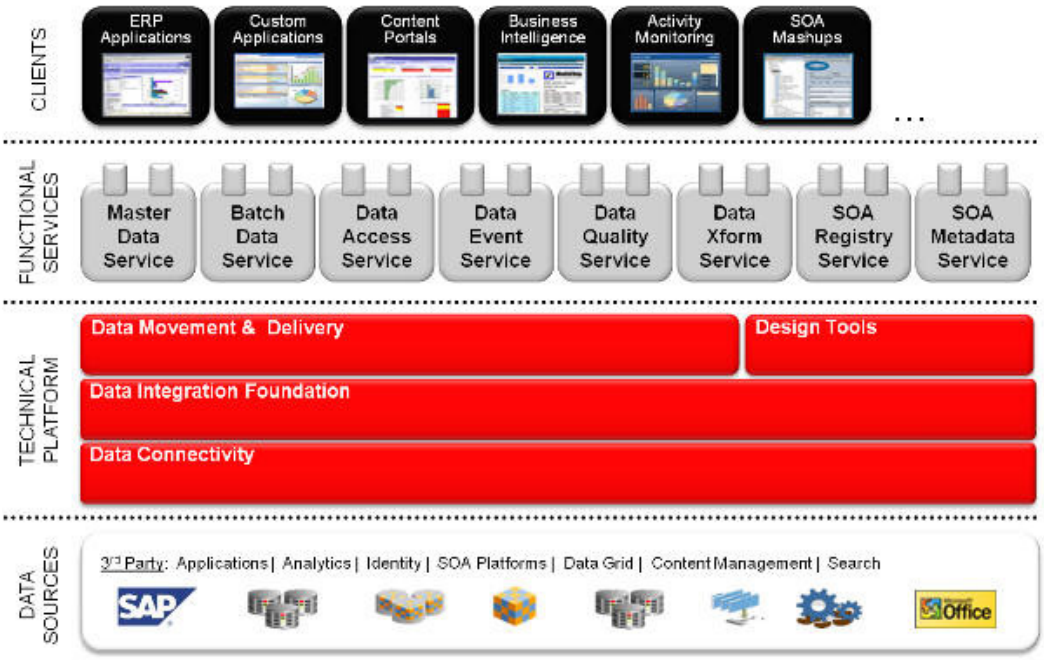


Figure 4: Enterprise Data Services Logical View

No single approach is the best for all possible Enterprise Data Services. And no single functional capability can fulfill all enterprise data needs. In the future of SOA, a hybrid approach for Data Services will dominate. A technical platform that can support these various delivery styles and functional services is now required to bring Data Services to useful maturity. (Figure 4)

Take for example a large financial institution that must simultaneously support high-demand, high-availability transactional applications, messaging integrations for thousands of application instances, and thousands of operational data stores and data warehouse grids. Traditionally these architectures would have each required fundamentally different infrastructures, from different vendors and with few overlapping solutions. But there is, and always has been one significant commonality among those diverse infrastructures -- the data. Core business data types like customer, product, order and others are connected across systems despite the relative isolation of enterprise infrastructure patterns. But why should they be?

A modern Data Service architecture should support synchronizing data grids with master data, publishing high quality canonical data within a messaging infrastructure, and expose control points for commanding BI and data warehouse systems as loosely-coupled services. This vision is not so much a dream, as it is a requirement for modern information-centric businesses that hope to use IT as a competitive edge within their industries. Yet regardless of how grand the IT strategy might be, a good Data Services plan will first solve fundamental tactical issues that simplify the use of data throughout enterprise architectures.

Business requirements and data architects will always demand a diverse range of Service Level Agreements (SLAs) that sometimes favor flexibility over speed, sometimes can operate in relative isolation, and sometimes require extreme availability, performance and scalability levels. Choosing the best mix of architecture, functional patterns, and delivery formats is essential for a rational, business-driven long-term Data Services strategy. Finding a single platform that can deliver this kind comprehensive flexibility should be on the short list of to-do items for any architect who is seriously exploring their Data Service alternatives.

About the Author

Mr. Pollock is a technology visionary and author of the enterprise software book "Adaptive Information“ (John Wiley & Sons). Currently esponsible for management of Oracle's data integration product portfolio, Mr. Pollock was formerly an independent systems architect for the Defense Department, Vice President of Technology at Cerebra and Chief Technology Officer of Modulant, developing semantic middleware platforms and inference-driven SOA platforms from 2001 to 2006. Throughout his career, he has architected, designed, and built application server/middleware solutions for Fortune 500 and US Government clients. Previously, Mr. Pollock was a Principal Engineer with Modem Media and Senior Architect with Ernst & Young's Center for Technology Enablement. He is a frequent speaker at industry conferences, author for industry journals, active member of W3C and OASIS, and formerly an engineering instructor with UC Berkeley's Extension for object-oriented systems, software development process and enterprise systems architecture.

More by Jeff Pollock

This content has been rated by 0 other people.
Average Rating: 0.00
Rate this content: 1 - Very Bad 2 3 4 5 - Very Good Vote!

More Top Stories

Understanding SOA Service Life-Cycle Management
SOA Needs a Bouncer
Web 2.0: Coming Soon to an Enterprise Application Near You
Your SOA is Only as Good as Your Relationship Triangle
What's Holding Up BPM Acceptance?
A Brave New Process Design Paradigm

More Top Stories
Feature Archive

Related News

CustomerVision Launches Enterprise Collaboration Service
Major League Baseball Selects Cognos for Stats Analysis

AIIM: Organizations Target Enterprise 2.0 Without Understanding the Market

More News

