

The SOA Magazine

Feature Article



Demystifying Data Federation for SOA

by Dain Hansen

Published: September 22, 2008 (SOA Magazine Issue XXII: September 2008)

[Download this article as a PDF document.](#)

Abstract: This article will unravel the mystery surrounding data federation. What are the essential requirements for a data federation solution? How is data federation considered a stepping stone to a successful SOA? Why are both data quality and data profiling so critical in bringing about manageable data services? What's the impact of these emerging federation techniques on data warehousing and business intelligence applications? This article will answer these questions through industry data, architectural patterns in data integration, and compelling case studies that illustrate the importance of embracing data federation.

Introduction: The Need for Reusable Data

In the struggle for information agility – take one part SOA and add it to two parts data, the result is the formidable and extremely popular data services. But what is it beyond the hype? Data services help transform data sources into reusable data components for the purposes of better *accessing*, *aggregating* and of *managing* data. Our SOA founding forefathers taught us that service-oriented architectures tend to be much more agile than their constrained EAI contemporaries. This is also the way we need to consider our data architectures.

There are a number of reasons that an organization might undertake a data-services implementation.

- Data is everywhere. The rate at which data grows in organizations is on the rise as is its critical importance for turning data into useful information.
- Data originates from a variety of structured and unstructured sources.
- Without data services, data access is too complex a task, involving different data models and data formats for each data source.
- The proliferation of point-to-point connections between numerous data consumers and providers increases as IT implementations evolve.
- The lack of a real-time, unified view across multiple sources makes data even more difficult to leverage consistently.

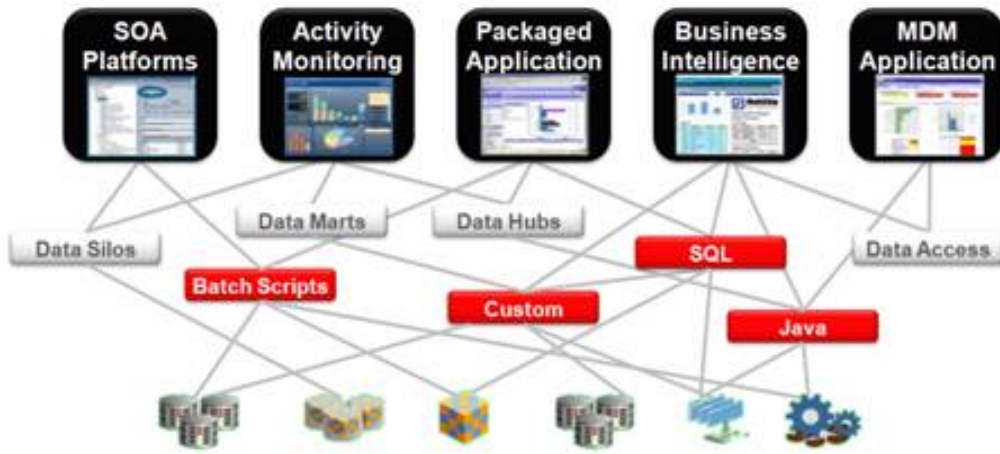


Figure
1

For most organizations, data appears on the architectural radar only when there is a mandate to derive specific value from collecting data. The data needed may be related to sales forecasts, campaign results, customer-service metrics, regulatory compliance, operational performance results, or customer profiling. Companies initiating projects that require data must make choices. For instance, should it continue to connect data in the same customized, rigid, point-to-point way that it uses for applications (Figure 1)?

Experienced IT professionals know that single-use data integration projects create long-term maintenance challenges and offer negligible ROI on the data integration portions of the project. The better strategy is to apply SOA principles to data integration, turning data into a service that is available as logical modules, each with a standards-based interface. This enables users to access and use the service more easily, improves data visibility, and promotes greater reuse.

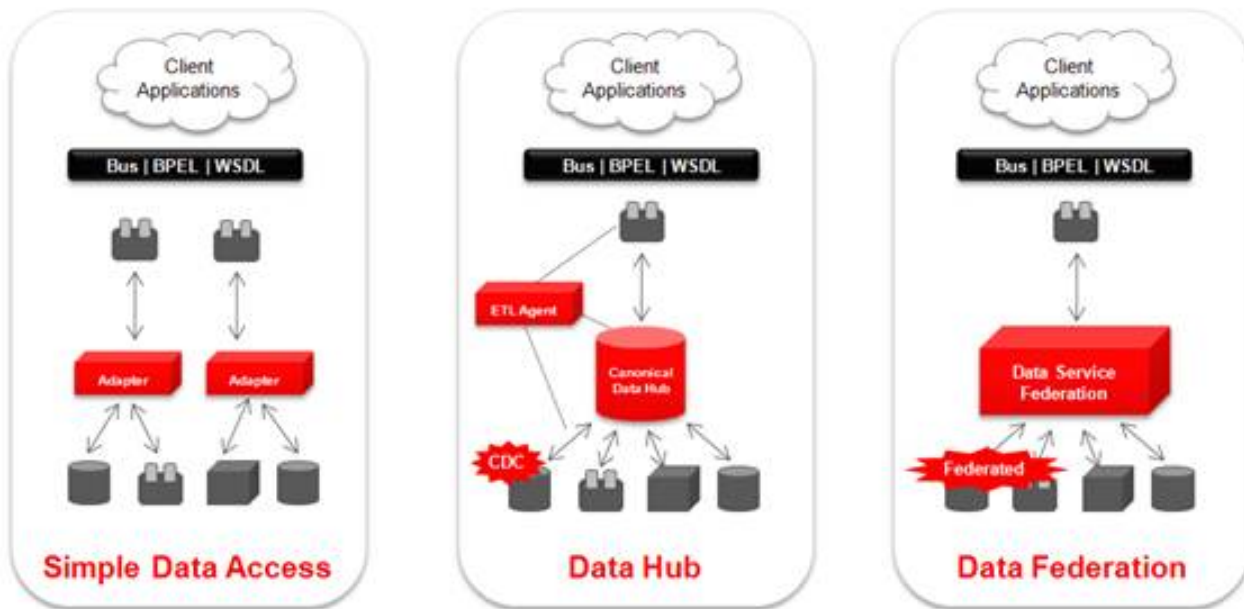
Understanding Data Federation Patterns

There is profound agreement in the industry that data services have a transformational influence on enterprise data-centric architectures. Our analysis indicates that there are three important scenarios where data can be exposed as reusable services (Figure 2):

- Simple Data Access
- Data Hub
- Data Federation Services

For a solution to meet expected results of the moniker of data services, it must achieve more than simple data access. Many off-the shelf SOA, BPM, DB products and even development tools include basic functionality for accessing data sources. Data services can also be generated from data-consolidation - bulk data scenarios through the use of ETL as a data hub.

Finally, data federation is defined as the capability to aggregate information across multiple sources into a single view. It leaves data at the source and consolidates information virtually, almost the same way an enterprise service bus (ESB) virtualizes messages but for data. Data federation essentially allows companies to aggregate data across multiple sources into a real-time view which can be re-used as a service.

Figure
2

We'll next look at these three examples in detail and discuss the pros and cons of implementing each pattern.

Simple Data Access

Build a data-source adapter into your SOA platform to access the data fields/records individually. Let your process application deal with putting the data together.

Advantage: Simple, low up-front costs, and easy to build

Disadvantage: Isn't highly reusable in and between large SOA implementations, leaves the manageability headache of mapping constantly changing data to business analysts and consumers of the data.

Data Hubs

Leverage bulk data transformation logic (ELT/ETL) for extracting, loading and transforming data into a consolidated data hub. This bulk data technique can be service-driven and also exposed for data access using the simple data service access approach.

Advantage: Essential for data warehouse and business intelligence applications, can scale for multi-terabyte implementations. Data hubs are ideal for large result sets or where such a consolidation provides optimizations for accessing and managing data. Also useful when data hubs can be managed offline from their original sources, for example analytics applications that should work off copied data from a web storefront.

Disadvantage: Requires copies of data which in turn require features like change data capture (CDC) to keep data in synch. The data hub approach might be overkill for smaller implementations. In some business cases there might be restrictions to copying data.

Data Federation

Aggregate data from multiple sources into a single view of the data and leverage that as a service to be re-used by your process application.

Advantage: Simple to build, reusable, all in one step, leaves the data aggregation mapping to a data services architect, leaves the data in place without requiring copying or synchronization logic.

Disadvantage: Performance characteristics must be closely watched and optimized because this approach adds an extra 'hop' or indirection through a federation server; the aggregation logic takes place in a middle-tier server instead of the database. Performance dimensions to consider include both the latency of the query transaction, and the load under large queries. The federation approach also introduces a new query paradigm (XQuery) which is more friendly

to XML data but might require new skills and training.

Each approach has its own appropriate usage in a data-centric environment. Let's now focus on data federation to explore its core functional requirements and what you should look for in evaluating a solution.

Functional Requirements for Data Federation

The basis for just about all service-oriented architectures is reusability. A high level of reuse can be achieved through a layer of data sources and business logic that are exposed as services accessible through an implementation independent service contract. A fundamental design goal of SOA is to provide a logical abstraction to these capabilities and prevent exposure of the physical topology of underlying sources to consuming applications. This is also the design premise of data federation and the basis for its functional requirements (Figure 3). In addition, a data federation solution should contain:

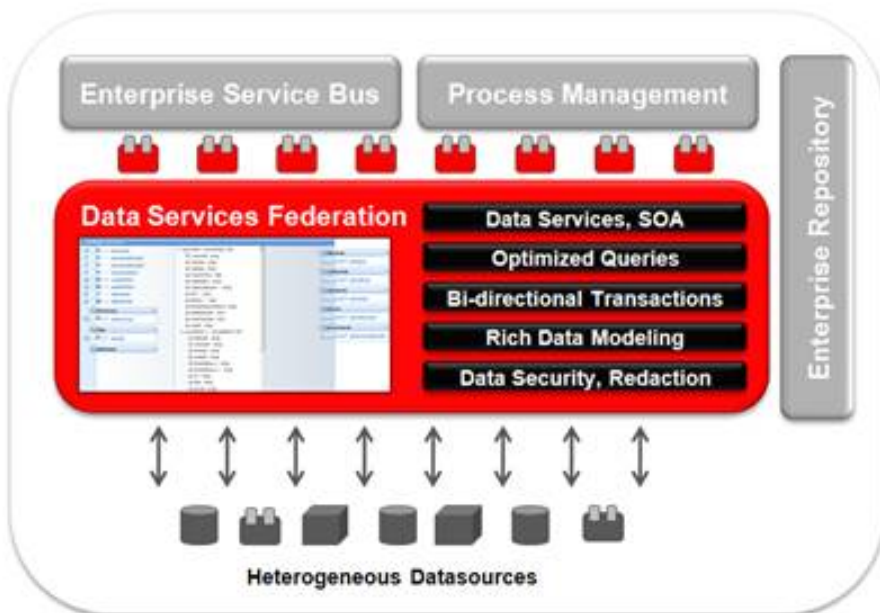


Figure
3

- Data Source Abstraction Layer for your SOA – Map multiple physical data sources into a set of services defined by logical entities rather than physical location. Reuse data services to define new abstractions as required.
- Federated, optimized Queries – Queries produce execution plans that use the specific capabilities of underlying databases and apply optimizations of distributed operations and nested data.
- CRUD-style Data Updates – Data services support creating, reading, updating and deleting of data in place at the original sources. Updates may be distributed and should support some type of reliability or XA compliance on the update cycle.
- Rich Hierarchical Model – Needs to be able to model relational data implementations as well as interchange with payloads from SOAP or REST services to be processed without format conversion (Figure 4).
- Security – Security poses new challenges for a data services environment: Services can be reused in unpredictable patterns, requiring flexibility while maintaining control of sensitive information. Data services require rich access control functionality, from policy-based authorization to fine-grained row and column-based security, identity propagation or credential mapping.

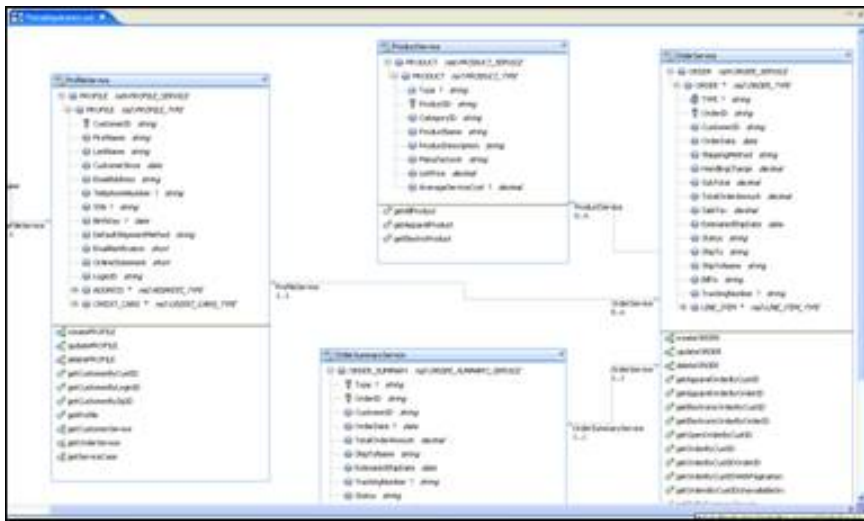


Figure 4

A data services federation layer is often seen as a way to take the first step in SOA. This services layer provides data mediation or abstraction between different data consumers and heterogeneous sources. Data services can be virtualized, aggregated views built from sources across the enterprise. They simplify data access and once created, are highly reusable. This approach eliminates the need to build workflows or code Java by hand, making it possible to automate data service creation and maintenance. Other consumers of data services include business processes, business intelligence applications, master data management (MDM), portals, and Web 2.0 applications.

Turning Bad Data Good

Bad data puts enterprise software projects at risk. Because inaccurate and inconsistent data exists pretty much everywhere, the demand for trusted data continues to spiral upward, driven by investments in packaged applications and business intelligence applications. Strategic IT initiatives like MDM also add additional pressure. Further complicating the matter, regulatory compliance initiatives (such Sarbanes-Oxley, U.S. Patriot Act, and Basel II) require tracing the source of the data used in financial reports, as well as examination, tracking (through snapshots), and certification of the state and quality of the business data.

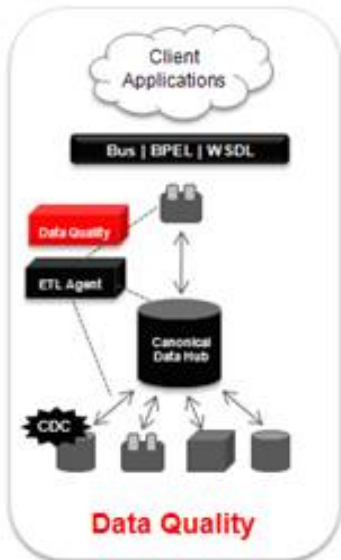


Figure 5

Technology alone doesn't deliver trusted data. Information managers need to define what data quality means to their organizations. These can be implemented through data governance programs that can define data quality rules and their respective processes for how they are maintained, approved and iterated to improve benefits.

Data quality generally requires data movement. Often we see data cleansing examples that are part of data hub approaches, as shown in Figure 6. That is not to say that data federation cannot implement a more real-time style of data quality during the process of data access, but the cleansing actions must be architected carefully for performance when they are implemented through the query access step of federation. For example, many MDM systems require a cleansed customer record to be presented on the application screen, if that record must be recovered from the source system without any cleansing; it can be performed in real-time once the data is accessed from the source. This type of real-time data cleansing is tricky to optimize correctly, but entirely possible in practice.

Regardless of the technology, without a fundamental data governance strategy at the core, these initiatives may suffer, just as many SOA implementations languished without a strong SOA Governance foundation to give greater visibility and control.

Consolidate or Federate... or Both?

At first glance it may appear that data federation and data consolidation are at polar opposites. Federation acts to abstract data from multiple sources into a view; consolidation acts to move data into a central hub. In fact, we're starting to see companies that use the best of both worlds together in their enterprise data-centric architectures. In fact, most pragmatic organizations start by integrating business services in a 'point-to-point' manner and only shift to a virtualization approach when the need arises.

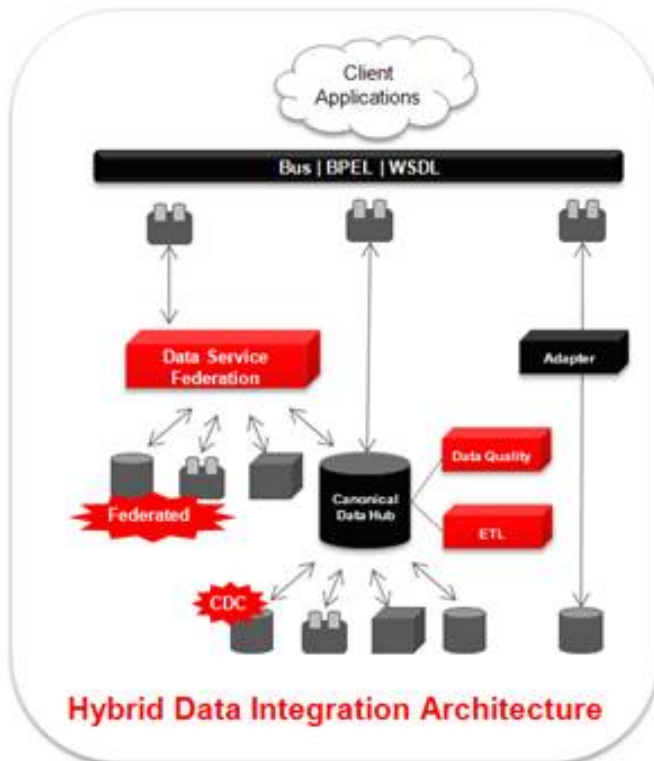


Figure
6

For example: A company may require a data hub for storing Web-store-front purchasing data. It uses consolidation/ETL to easily build, deploy and manage this data hub. It can change data capture technology to keep this data hub in sync with any real-time updates. But for a call center application that needs customer information and a whole lot more, it may need to access data beyond the single customer hub. The data used by the call center may live also across other data marts and other application constituents, hence an opportunity for data federation as an alternative to designing a new call center operational data store (Figure 6).

An exciting element of this platform is that vendors are merging different data integration technologies into unified solutions, enabling multiple tools to be used at will. We will undoubtedly be seeing more of these approaches as federation becomes useful for business intelligence and MDM applications that are broadening the need for more flexible and agile data integration approaches.

Case Study: Financial Services Using Data Services

One global financial organization struggled with the continuous flux of merger and acquisition activity in which data integration was a perpetual challenge. In this example, disparate subsets of information prevented the organization from achieving a single, organized customer view. This impacted their intimacy with the customer and slowed development of improved financial service products from being developed. Because their customer insights were fragmented, consequently their responsiveness to customers diminished as well.

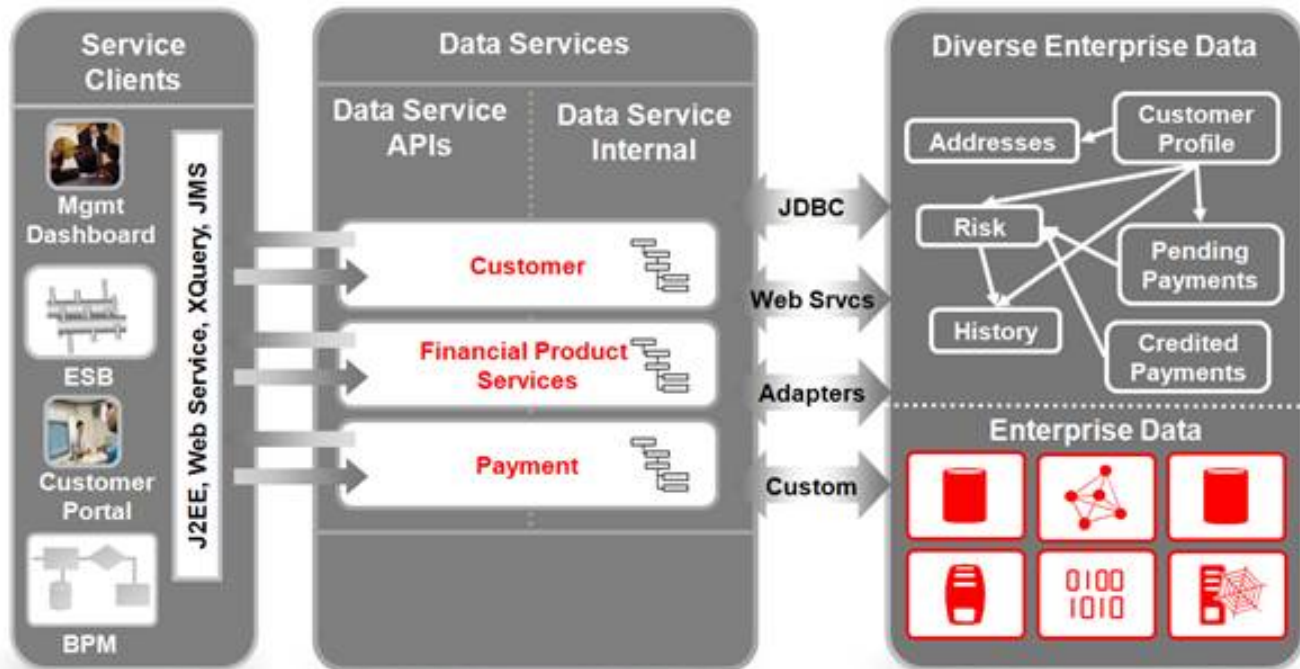


Figure 7

The organization was challenged to correct these gaps by establishing a data-services layer to reconcile data, alleviate the demand on database administrators (DBAs) and developers, and reduce time to value of new financial service offerings. This organization had started down the path of simple data access using home grown mechanisms but quickly realized that the solution wouldn't scale, was hard to manage and didn't perform. In addition the organization didn't want to re-invent the wheel of their data-tier and leverage existing data hubs that had already consolidated most of the company data.

Their goal: Develop highly flexible, scalable single views of the customer, based on a data across fragmented data sources.

Their methodology: Federate data within a Web services abstraction layer for better views of customer information. Leverage the existing data mart as the source.

The following issues were considered:

- Physical changes to the databases have a dramatic impact on Web services, these needed to be abstracted out so that they didn't impact the consumers of the service.
- Implement the data services as an abstraction, not as a replacement of their existing data marts and data hubs.
- Data services must scale without performance degradation.
- The data services solution must preserve or improve on current service-level agreements (SLAs).
- The original data services had originally been ineffective because each line of business treated a customer as separate customers with separate interface requirements. Multiple customers and partners needed to be supported in a single interface.

- Other business units had refused to adopt the original Web services interfaces because the data implementations were rigid and inflexible. These implementations needed to offer agile service bindings that could change quickly, offer new data signatures (the technical contract), be quick to model and implement, while still conforming to different groups, and business requirements.

The Results

Data Services using the federation technique reduced overall complexity with greater data consistency and reusable data services. These services initially became extensions to the data tier implementation and later became adopted as new models within the data warehouse itself. This type of hybrid model reduced development from 12 weeks to 80 hours, and now requires half the number of development managing and maintaining the implementations. No changes to the database were required, freeing the DBA teams to address other tasks. User access to data no longer requires communication with back-end systems, preserving performance within SLA restrictions. The ROI extends beyond this individual project, freeing this growing financial organization from constraints that might otherwise affect its customer intimacy and its overall future business.

Conclusion

There are different approaches to building data services. Data federation provides an important option that results in a rich abstraction layer aggregated across multiple sources. Nevertheless, this approach shouldn't be considered in a vacuum without weighing carefully the business objectives and the implementation requirements. Consider first the options of consolidation, federation, or perhaps using both together for improved agility and manageability. Regardless of your endeavor you'll most likely need to consider a data management strategy for ensuring consistency, accuracy, and better control of your data implementations. Data services are likely to continue to evolve as data warehousing, SOA, and business intelligence technologies continue to mature.

THE PRENTICE HALL SERVICE-ORIENTED COMPUTING SERIES FROM THOMAS ERL



[Home](#) [Past Issues](#) [Contributors](#) [What is SOA?](#) [SOA Glossary](#) [SOA School](#) [SOA Books](#) [About](#) [Legal](#)

Copyright © 2006-2008
SOA Systems Inc.