



PATH TO REUSABILITY

Using governance and service architecture to reduce unmanageable service proliferation

Abstract: In an attempt to simultaneously enhance business agility while dealing with the increasing complexity of IT needs, businesses have turned to service orientation to build out IT capabilities. But instead of achieving the touted cost savings from service reuse, services have turned into standalone applications that have to be maintained individually. These services proliferate the enterprise and bring with them typical development difficulties. MuleSoft recommends a multi-tiered approach to reexamine enterprise service orientation and increase reuse from typically less than 10% to a goal of over 50%.

Jamie Triplett
Solution Architect
jameson.triplett@mulesoft.com

www.mulesoft.com | info@mulesoft.com | 1-877-MULE-OSS

1	What's the Problem?	3
2	Organizational (Top-Down).....	3
2.1	Center of Excellence (COE).....	3
2.2	Project Governance	4
3	Technical (Bottom-Up).....	5
3.1	Services Architecture	5
4	Final Thoughts.....	7
5	How Mule Does This Differently	7
6	Developing Your Solution	8
6.1	Phase 1: Assessment	8
6.2	Phase 2: Proof of Concept.....	8
6.3	Phase 3: Transition	8
6.4	Phase 4: Support	8

1 What's the Problem?

IT environments are constantly changing. Business, customer, and data needs change over time—sometimes very quickly. Last year's business critical apps become this year's ho-hum. With each new application that an organization brings in, a flurry of new development—typically entirely custom built—takes place. Usually, with these new applications, less collaboration takes place across silo lines. All businesses desire a dynamic IT environment that responds to changing needs and without some organizational and technical frameworks, services proliferate within the environment and become very chaotic and difficult to manage.

In an attempt to simultaneously enhance business agility while dealing with the increasing complexity of IT needs, businesses have turned to service orientation to build out IT capabilities. Using the monolithic development paradigm that was effective with previous problems leads to a proliferation of services and each new application requires new, complex, and specific services to be built. In this paradigm, developers create new work instead of reusing old work because subtle differences exist between previous and current needs. Typical default difficulties arise in services development such as scant documentation, “copy-paste” code reuse, and versioning. Services become little more than one-off components that cannot be reused.

MuleSoft recommends a multi-tiered approach to service proliferation within an enterprise. Organizationally, we look top-down, review the enterprise service architecture and IT governance model and create a Center of Excellence for both expertise and standards enforcement. On the technical side, we take a bottom-up approach: structuring development patterns, separating application and data layers and reviewing the development processes. This results in an increased reusability of built services and a loosening of the coupling between elements. The MuleSoft approach aims to boost the typical 10% reuse rate to 50% or better.

Top Down: Organizational

- Enterprise Service Architecture (ESA)
- Center of Excellence with Experts and Enforcement
- IT Project Governance, and Review Processes

Bottom up: Technical

- Development Patterns
- Data Model
- Standardized Data Services
- Application Service Layer

2 Organizational (Top–Down)

Enhancing reuse and reducing proliferation requires an Enterprise Service Architecture (ESA). Although this sounds complex and difficult, its few basic components can be implemented without too much difficulty. Reusable services must be governed as shared assets in order to maintain and access them. The first basic need, a Center of Excellence (COE), must have expertise on available services, services architecture, and standard patterns. Secondly, a level of governance is necessary to ensure projects involve the COE and enforce its decisions.

Implementation of the organizational changes necessary to keep service proliferation to a minimum requires most importantly the endorsement of the CIO or head of IT. The necessary organizational changes include creating a cross functional body independent of any department's budget, making slight process changes when services are developed, and enforcing the new policies and processes across IT.

2.1 Center of Excellence (COE)

The COE is a cross functional group with members from IT leadership, IT architecture, and development. Primarily, the COE assists with the development, reuse, and enforcement of services, and service standards.

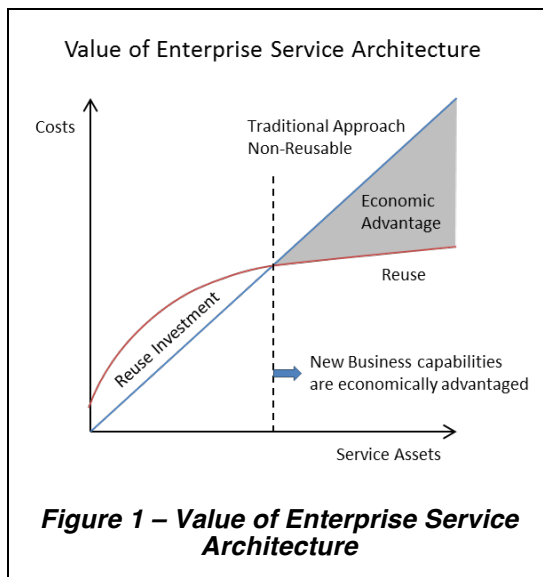
To enhance reuse, the COE must have expertise and ownership of the enterprise services that projects develop. During the initial stages of a project, when the architecture and design of the project are described, the COE is involved in any service implementation decisions and research. The COE also reviews designs of new services and either approves or redesigns them. The COE develops and owns the ESA, which is a business, process, and technology architecture that must build and maintain a successful shared-services infrastructure.

The basic components of a COE are:

- People
 - Chair: makes final decisions, heads meetings, promotes services
 - Architect: designs and reviews service architecture, researches across department lines
 - Librarian: maintains service registry, checks in services to the registry
- Process
 - High level architecture/solution: at project initiation when requirements and designs are being discussed, the COE proposes architecture for using and building shared services
 - Review: the COE reviews designs before and during implementation
 - Publish: after implementation when the services move from development to production, the COE checks in services and hands them off
 - Changes: the COE manages changes as they arise and makes notification of changes
- Technology
 - Registry: a services registry is a critical component to centrally locate and document services that are available, and what uses them
 - Shared services hardware or cloud infrastructure: the hardware that services run on, or cloud contracts/infrastructure

2.2 Project Governance

The COE plays a distinct role in the project lifecycle that must be enforced in order to realize its value. To fully succeed, the COE is integrated into multiple stages of project development: at the initial phases of discovery and architecture, at the design phase where details are finalized, and at development time for expertise and support. To integrate the COE, project governance and standard processes must be updated.



An enterprise moving toward a new ESA will face the task of addressing the cost concerns with shared services that cross department lines. Most projects are measured by time and budget. Building enterprise quality reusable services in general takes more time and budget than building non-reusable services. The savings do materialize down the road however as those reusable services are implemented into subsequent projects, saving substantial time and effort. Discussing an exact solution falls outside the scope of this paper as solutions vary highly and depend on enterprise structure.

At the initiation phase of a project, some considerations will frame the shared services discussions around time and cost. Understanding these tendencies and building in a system in the enterprise to deal with them will enhance the adoption of shared services and the processes that build them. These considerations include the following:

- Which budget pays for building shared services?

Building enterprise grade services accrues extra costs and departments with limited budgets will hesitate to foot the bill.

- If the department does foot the bill, are there incentives for building useful services for enterprise reuse? These can come as published metrics and charge backs.
- If a shared infrastructure budget is in place, what does it cover? Does it cover the shared service implementation portion of a project? Does it share the cost of the entire project if shared services are implemented?

In the longer term, other considerations come to the fore past the first implementation of the service. The service must be maintained, supported, and occasionally changed, and the budget must be allocated to support the service over time. Without some set of incentives, departments alone will not want to foot the bill for ongoing, shared-services line items. These expenses can include:

- Costs related to hardware or service providers that the service operates on (e.g., the cloud, or an in-house server)
- Hardware upgrades as a result of the heavy use of a very popular and successful application
- Occasionally the service may need modifications to accommodate changes in technology, the marketplace, etc.

MuleSoft recommends that enterprises handle these issues by developing a services COE budget both for implementation and maintenance. Allocated funding allows departments to get what they need by dipping into the shared services pot and saving their own money, effectively getting a piece of their implementation for free. This usually requires a high-level backer to secure the budget necessary to support fully a shared-services infrastructure.

3 Technical (Bottom–Up)

As new services are developed, teams look to previous implementations for ideas, options, and inspiration. They tend to then develop something new that may look similar in functionality but does not actually reuse any previous work. This natural aversion to re-inventing the wheel entirely results in a new build, customized for a particular need.

At first look, taking inspiration from an older service seems to be an easier, low-hanging-fruit type of solution. And it certainly saves time over creating something entirely new. But keeping the developed service highly specific and complex misses a substantial opportunity at reuse and adds to the problem of proliferation of services. Preventing this proliferation requires three fundamental technical foundations: a clear data model, data services that access that data model, and application services that orchestrate the foundational components into specific uses. In the previous development paradigm, these two separate service layers are typically packaged into one large program.

3.1 Services Architecture

In a well-built, de-coupled services architecture, distinct layers each play specific roles in mitigating service proliferation. In addition, each service is relatively simple and easy to maintain and unit tests are well specified based on service standards. In Mule, services are built as Mule flows.

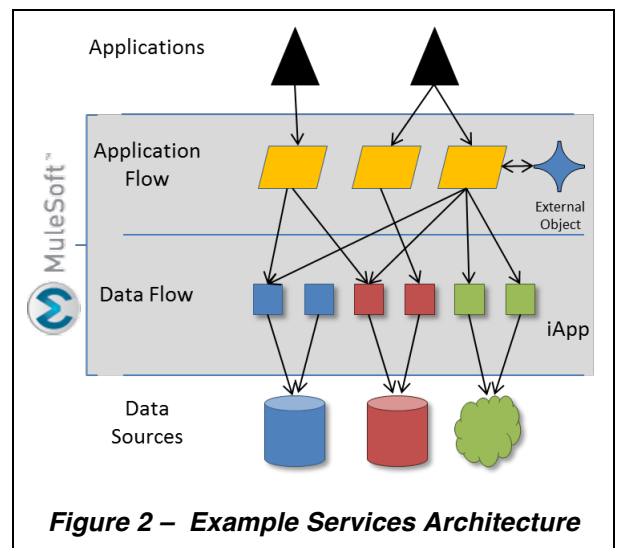


Figure 2 – Example Services Architecture

The bottom layer in a service architecture is the enterprise reference data sources that feed data services. These can be databases, mainframes, cloud applications, or any other internal/external reference source. Typically these sources have standard APIs that Mule Connectors integrate with. In the cloud case, Mule iON is a perfect integration solution.

Data Services are the foundation for basic Create, Read, Update, Delete (CRUD) operations on the data sources. The data model underlying the data source and its relationship to other sources specify these data services. As much as possible, these data services should be designed from the data model and built to enterprise standards—not customized for a project. In some cases more complicated operations can be implemented as data services if they are standard operations performed by multiple applications.

Above the data service layer sits the application service layer, containing the customized services that call one or more data services and objects. External objects in these cases can be custom built components, business rules engines, or anything else. Application services are built separate from data services for two primary reasons:

- To enhance reusability of the standardized data services by keeping any customizations outside
- To increase maintainability of application services and data services; if and when things change, the separation will allow much easier change control; separation also simplifies dependencies and reduces substantially the amount of “cut and paste” code development

Additionally, de-coupling, application services, and data services should not need any knowledge of each other’s internal workings. Application Services will be built for each project and the portfolio will expand as projects are built however the flow itself can be retired as the application it serves is retired.

Although data services and application services are developed in the same project to service one application, their purpose and scope are entirely different. Data services become an enterprise reusable asset; application services are the custom work for each application. It’s unrealistic to believe that all the development for one application can be reused. As previously stated, the goal is to go from 10% reuse to 50% or better reuse by building standard and reusable components and services as part of a holistic solution.

4 Final Thoughts

The primary purpose of an IT department is to enable businesses to do things better, faster, and cheaper. Enterprise standards and reusable services and services address all three of these major goals. A strong reuse goal increases the agility of a business and reduces the overall costs of building out functionality.

The chart to the right demonstrates the importance of taking a strong approach to reuse. As the reuse percentage increases, the breakeven point for an organization moves forward, though initially reusable services cost more than custom ones. Many organizations can get stuck at a low reuse percentage and never realize the cost savings. An aggressive program of governance and technical architecture, patience to see the program through, and good metrics to measure the progress of an enterprise program are required to reap the gains. Without these practices, any efforts to reuse services will not succeed in the long run as projects will default to the cheaper, faster approach.

This paper has covered the basic requirement for reducing the service proliferation that can create complex and difficult to maintain infrastructures. All of these approaches must be examined in the context of your enterprise and your specific needs. MuleSoft is proud to offer a program to assist your enterprise with its services, services, and organizational governance.

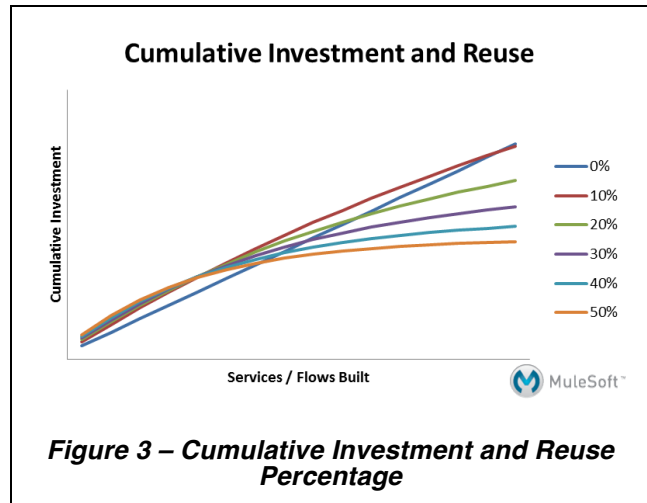


Figure 3 – Cumulative Investment and Reuse Percentage

5 How Mule Does This Differently

What makes MuleSoft different from any of the large vendors in this space all touting governance, reuse and Service Architecture? MuleSoft offers a multi-level approach, most SOA initiatives are done from either the bottom or the top. Top down approaches use big vendors and consultants to create big-bang projects. Bottoms up approaches tend to miss the cross-functional potential of real reuse.

From a top-down budget perspective, Mule has several key advantages. Mule has all the enterprise functionality one would need and expect in an SOA platform while being exceptionally lightweight. It also delivers a faster time to ROI than other solutions by providing a great deal of architectural flexibility. Unlike other solutions which often require a “big bang” approach, where several moving pieces must be deployed at once, Mule doesn’t impose architectural restrictions, allowing Mule customers to take a phased approach: first focusing on their most pressing business needs and then incrementally moving towards a final vision. Furthermore, since Mule is lean and mean, requiring much less processing power and memory footprint than other approaches, the upfront hardware investment is substantially more modest than other approaches.

Mule is an especially developer friendly platform. Mule offers fast development and assembly of services and components, with an incredibly flexible model allow developers to solve problems in multiple ways and use their creativity. Mule also takes advantage of tools developers are already using, such as Eclipse, Ant and Maven, rather than requiring adoption of a new set of idiosyncratic development tools. This means developers can more quickly embrace Mule and less investment is needed in training. Finally, as cloud services become more pervasive, Mule’s Cloud Connector capabilities and iON Platform as a Service give it best in class support for these new services, greatly simplifying integration development with these services.

6 Developing Your Solution

MuleSoft is excited to offer a package to assist your enterprise with its Service Architecture and Service Governance. Our approach is a multi-phase to get the best value, incrementally building upon an initial assessment towards a proof of concept to demonstrate value, and finally transitioning project leadership to your enterprise.

6.1 Phase 1: Assessment

In this phase a solution architect is assigned to your organization to assess the current service ecosystem. Where your services are, how many are there, is there a proliferation problem, and how best to apply best practices. At the end of phase 1 there is a plan for a phase 2 proof of concept using the practices above.

6.2 Phase 2: Proof of Concept

We take the planning of phase 1 and develop a proof of concept for your organization showing the best practices in use. This project may run through the entire project life cycle in your organization with MuleSoft leadership. The important parts are that the project is run through organizational elements like the COE, and developed to your organization's Enterprise Service Architecture.

6.3 Phase 3: Transition

Building on the success of the Proof of Concept, the transition phase is a deeper dive into your services eco system to look at where transition work is need from previous implementations. This phase goes into depth examining the pain points around spaghetti architecture, and launching a new architecture successfully. Projects and service architecture is transitioned to be led by your enterprise and/or by a MuleSoft development partner

6.4 Phase 4: Support

As needed support is always available for your Mule infrastructure as well as any project needs that may come up. Your organization will have a functioning Services COE that may be augmented occasionally by a MuleSoft Solution Architect or development partner.

Jamie Triplett is a Solution Architect at MuleSoft. He works with MuleSoft Services customers to develop solutions for their business and technical needs. Jamie has been working in enterprise architecture and service oriented architecture (SOA) for four years. He's been involved in multiple engagements around building services, services governance, and enterprise architecture.

About MuleSoft

MuleSoft is the Web Middleware Company, providing enterprise-class software built on the world's most popular open source application infrastructure products, Mule ESB and Apache Tomcat. With Mule ESB and Teat Server, MuleSoft brings an ideal combination of simplicity and power to today's web applications. Its products boast more than 1.5 million downloads and over 2,500 production deployments by leading organizations such as Walmart.com, Nokia, Nestlé, Honeywell and DHL, as well as 85 in the Global 500 and 5 of the world's top 10 banks. MuleSoft is headquartered in San Francisco with offices worldwide.

For more information: www.mulesoft.com, or email info@mulesoft.com.

Download Mule ESB: <http://www.mulesoft.com/download/>

MuleSoft and the MuleSoft logo are trademarks of MuleSoft Inc. in the United States and/or other countries. All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only.

All contents Copyright © 2011, MuleSoft Inc.