

**RED HAT  
SUMMIT**

**LEARN. NETWORK.  
EXPERIENCE OPEN SOURCE.**

June 11-14, 2013  
Boston, MA





# Loosely Coupled, Reusable & Interoperable

## JBoss Integration Overview and Roadmap

**Keith Babo**  
**Jack Britton**  
**Alan Santos**  
**Red Hat, JBoss Product Management**



# Agenda

JBoss Integration History

Product Line Overview

Q&A

# JBoss ESB

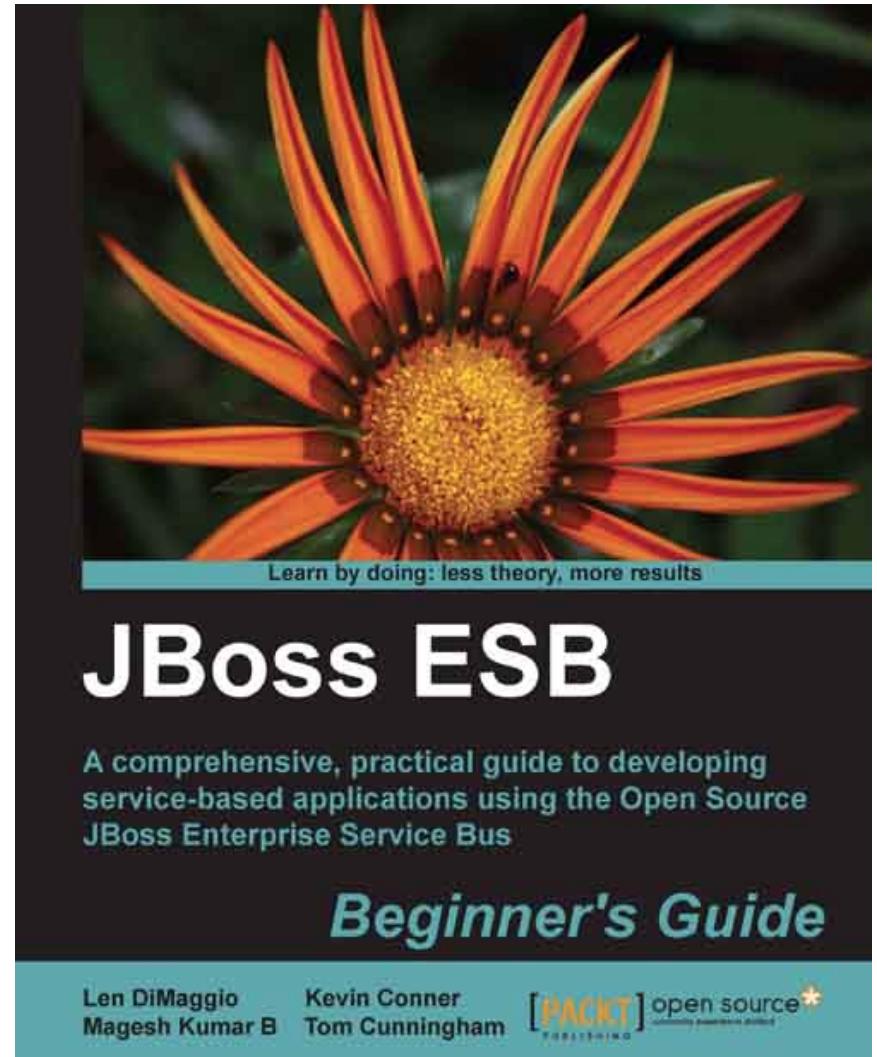
Red Hat SOA Platform 4-5 based on  
“JBoss ESB”

Well adopted, proven and functional

Non-standard design

Steep learning curve, limited tools

Difficult to effectively monitor/manage



# JBoss Integration 2012

## Customer Feedback

“Reduce complexity and learning curve”

“Improve monitoring/visibility”

“Difficult to maintain”

## 5.X Technical Direction

SOA 5.latest – Added Camel Gateways, Improved Monitoring

## Fuse Acquisition

### Red Hat to Acquire FuseSource

Raleigh

June 27, 2012

Open source integration and messaging provider will enhance Red Hat's JBoss middleware

RALEIGH, N.C. – June 27, 2012 – Red Hat, Inc. (NYSE: RHT), the world's leading provider

# JBoss Integration 2012

## Product Goals

Accessible,  
Service Centric  
Operational Control/Visibility,

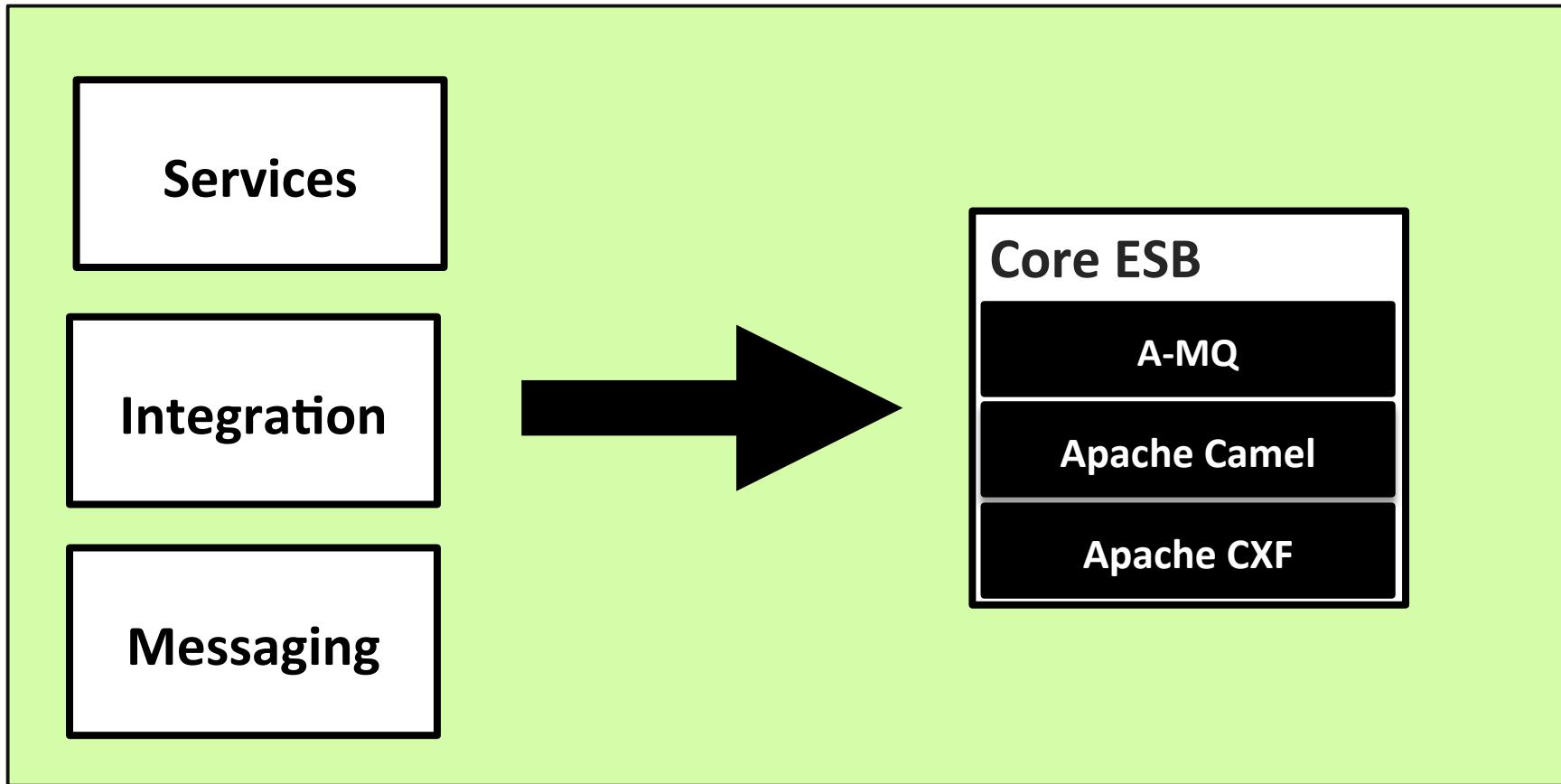
## New Users

FuseSource acquisition

## New Technology

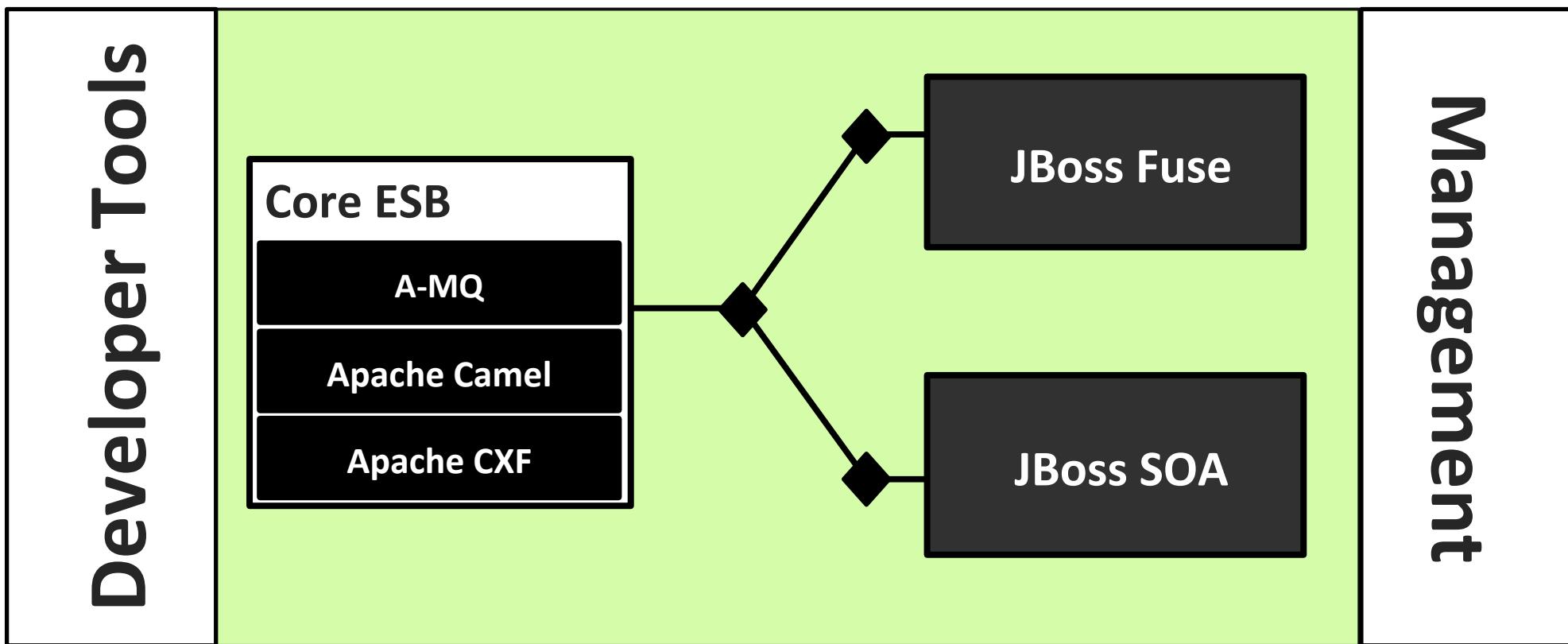
Light Weight, Fast, Manageable

# A new ESB foundation



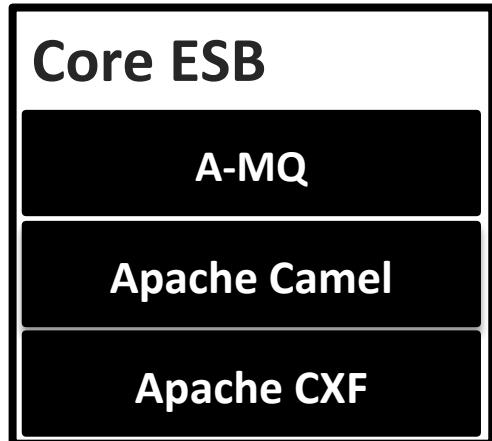
“...a software architecture model used for designing and implementing the interaction and communication between mutually interacting software applications...”

# Fuse product line

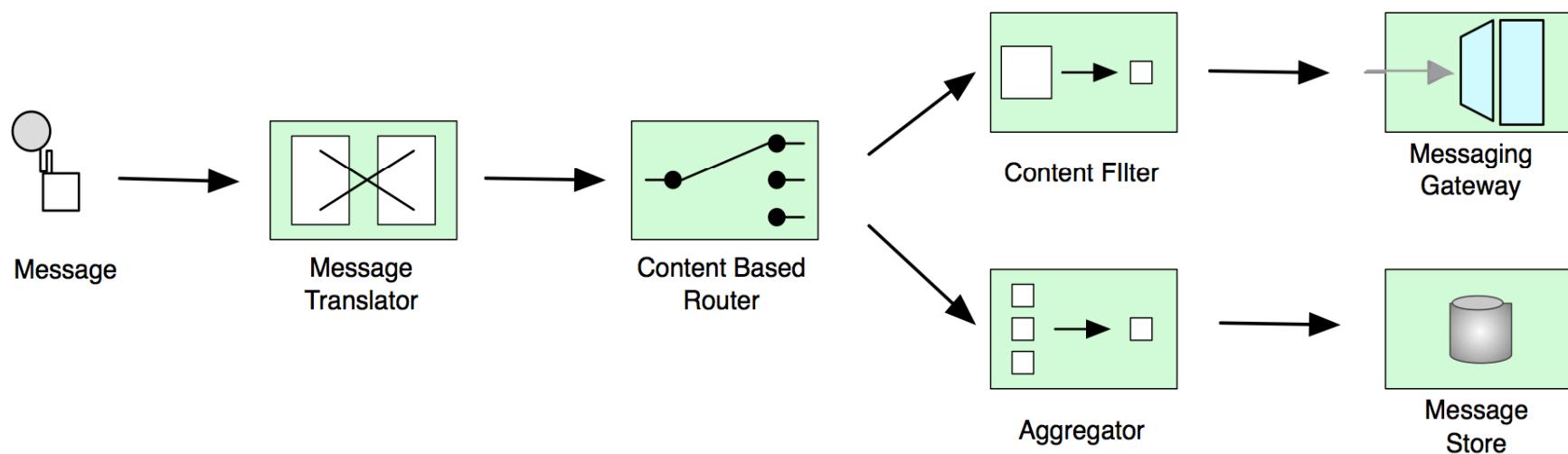


1 ESB, 2 products, tiered functionality

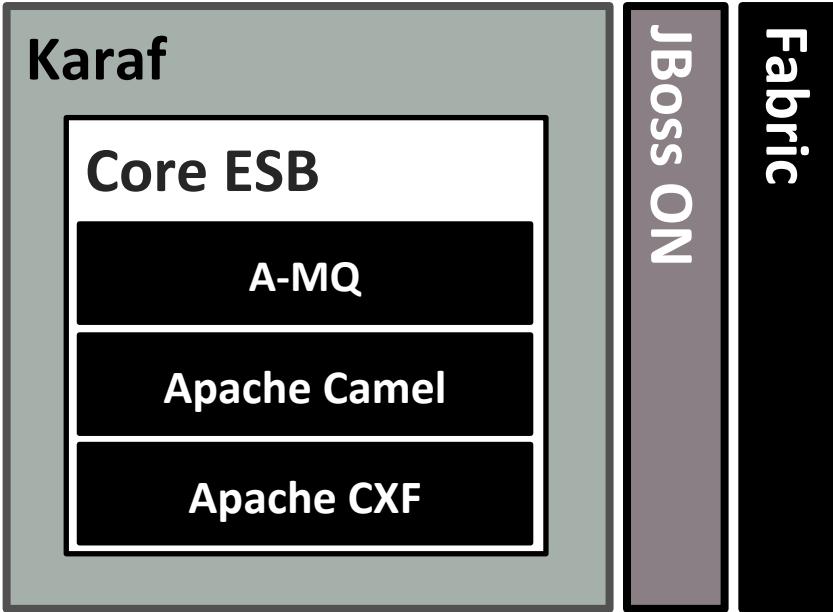
# Core ESB



Simple, extensible service framework



# JBoss Fuse



The de-facto, open source ESB

Apache technology and license

EIP as a first-order concept

Simplifies  
Integration, Transformation, Mediation

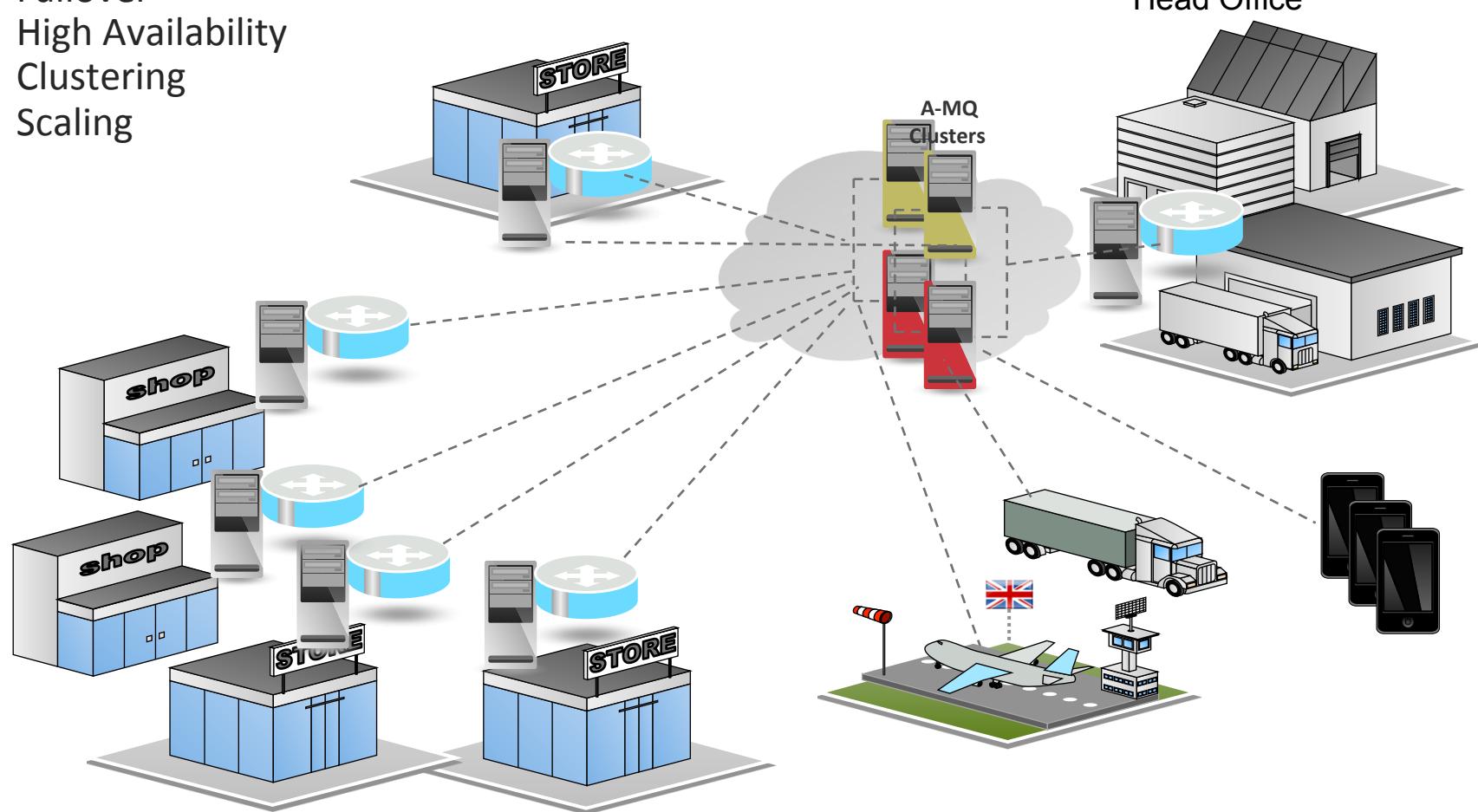
Built for developers, emphasizing code/configuration

OSGI

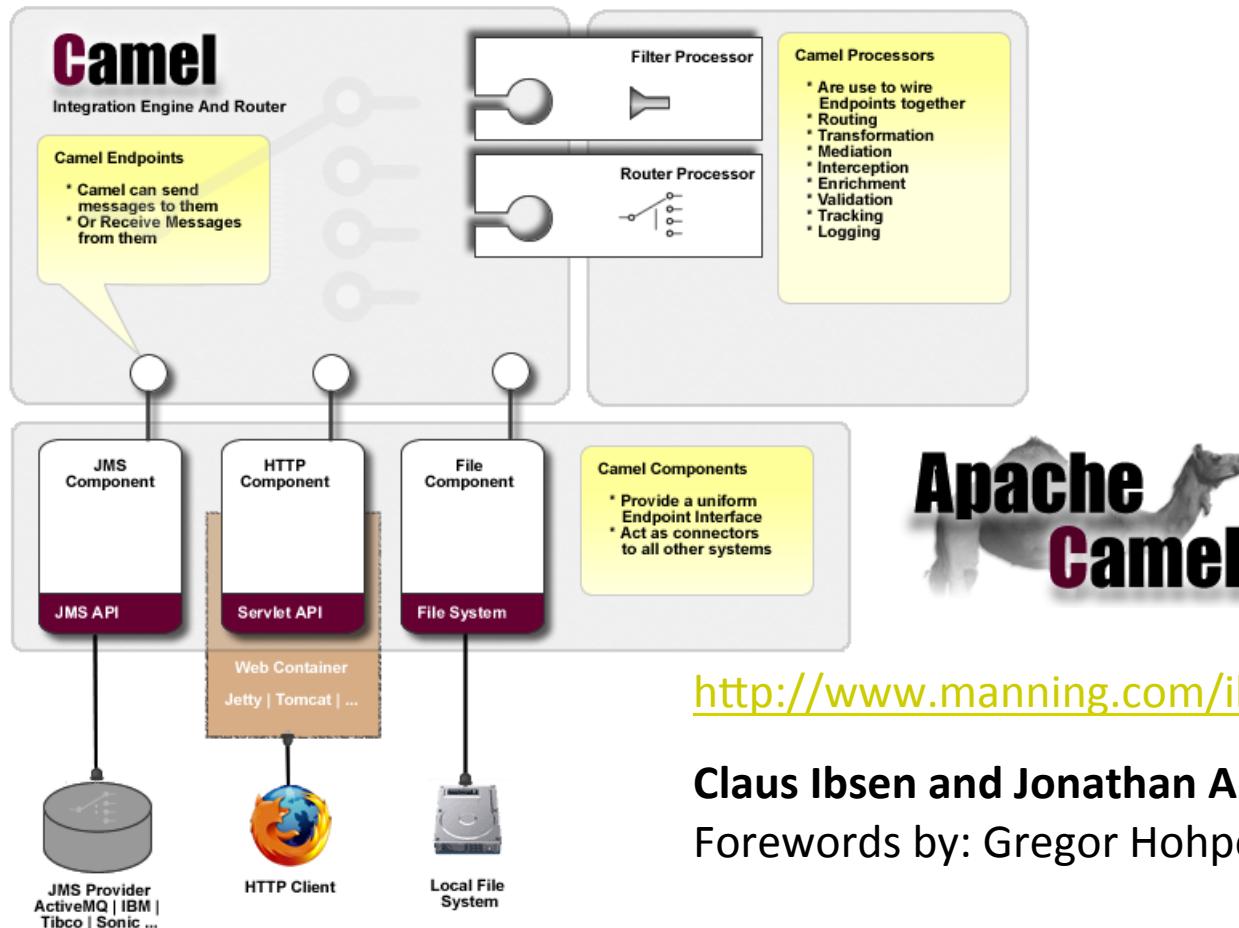
Management via Fabric and JON

# Core ESB - ActiveMQ

- Failover
- High Availability
- Clustering
- Scaling



# Core ESB - Camel



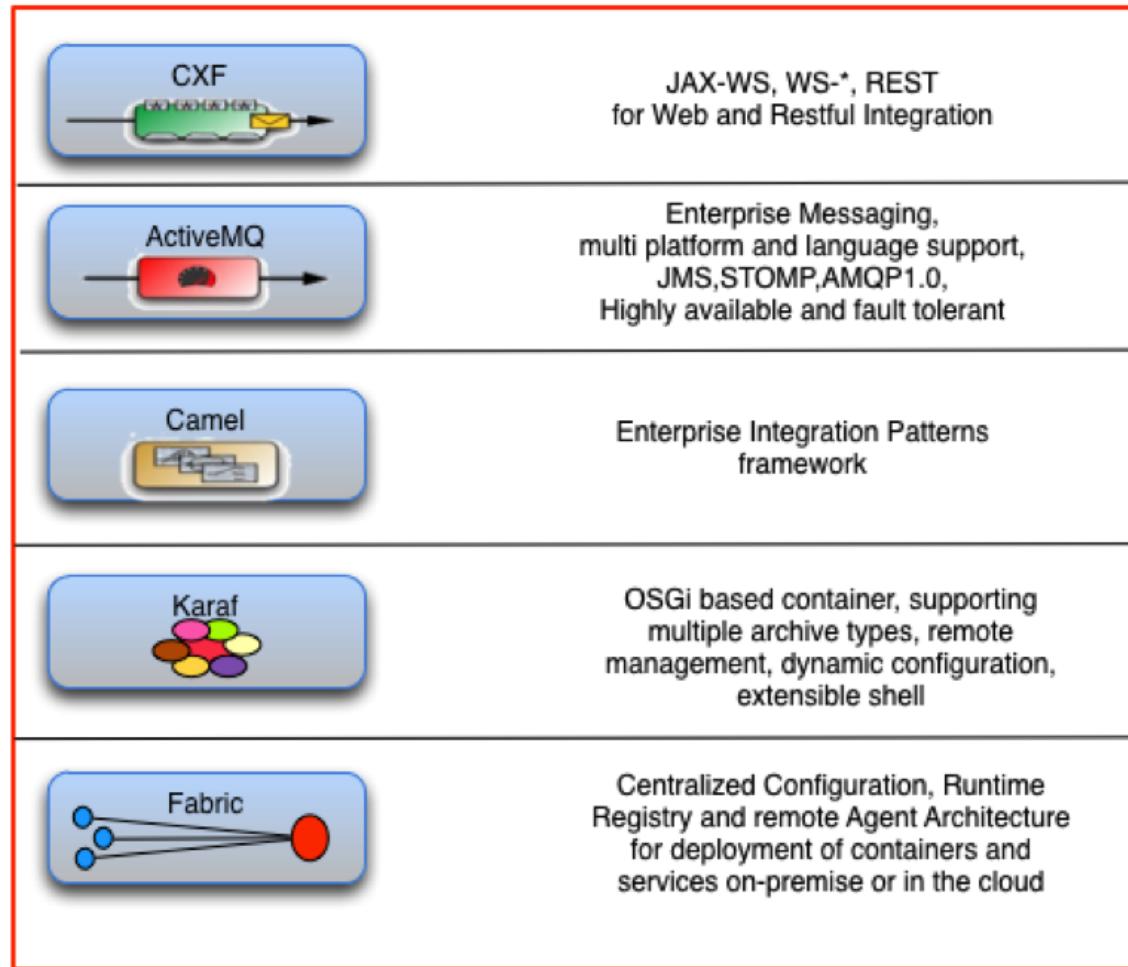
<http://www.manning.com/ibsen/>

**Claus Ibsen and Jonathan Anstey**  
Forewords by: Gregor Hohpe and James Strachan

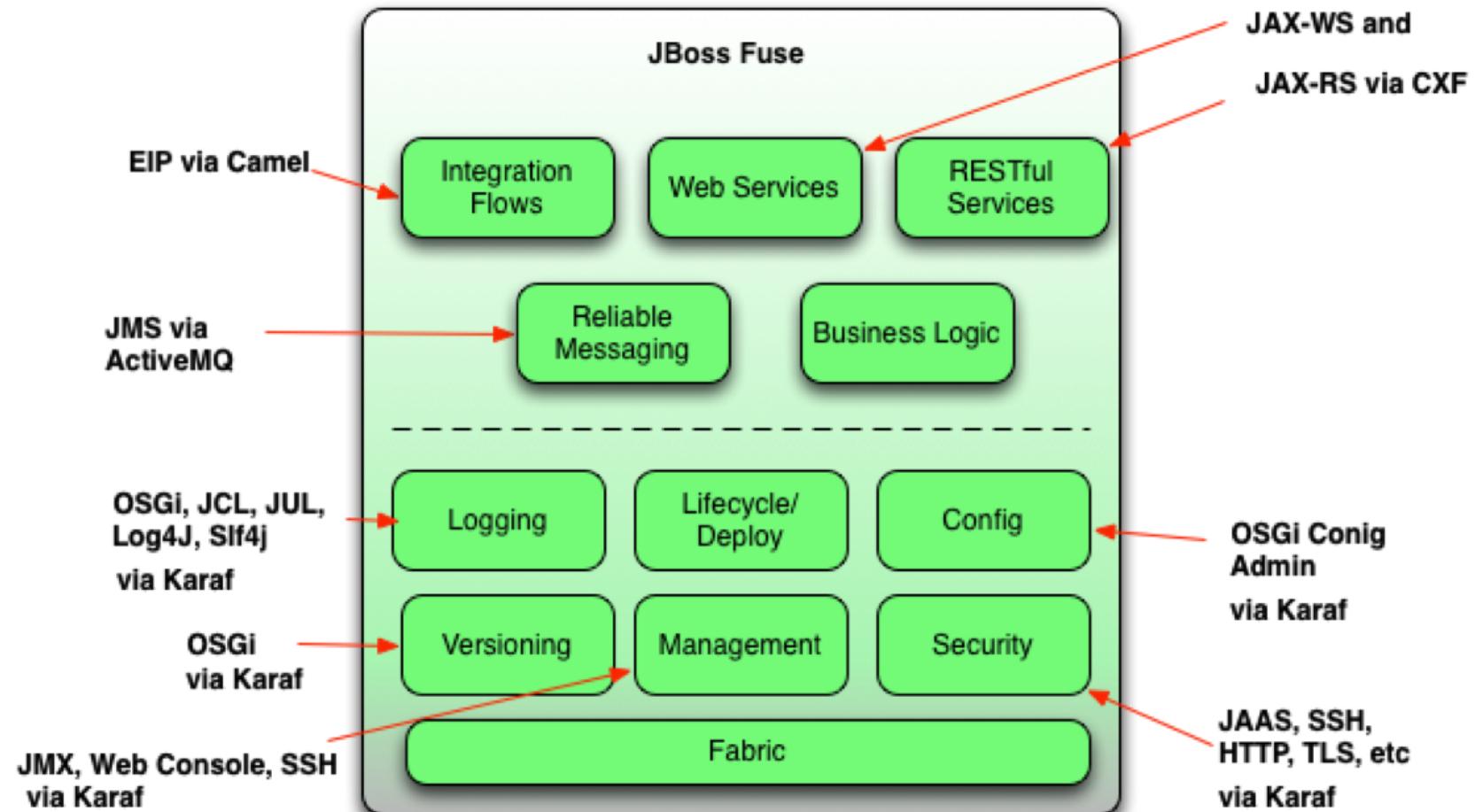
# Core ESB - CXF

- Leading open source web services stack for Java
- Most flexible approach
  - Configure using annotations
  - Manipulate configuration directly in code
  - Great integration with Apache Camel
- Very active development team, fast bug fixes and enhancements  
JAX-WS and JAX-RS Certified
- DOSGi reference implementation of OSGi Remote Services Specification
- Used by FuseSource, JBoss, WSO2, Paramiti, MuleSoft, Talend, TomEE etc.
- Used by Google, TomTom

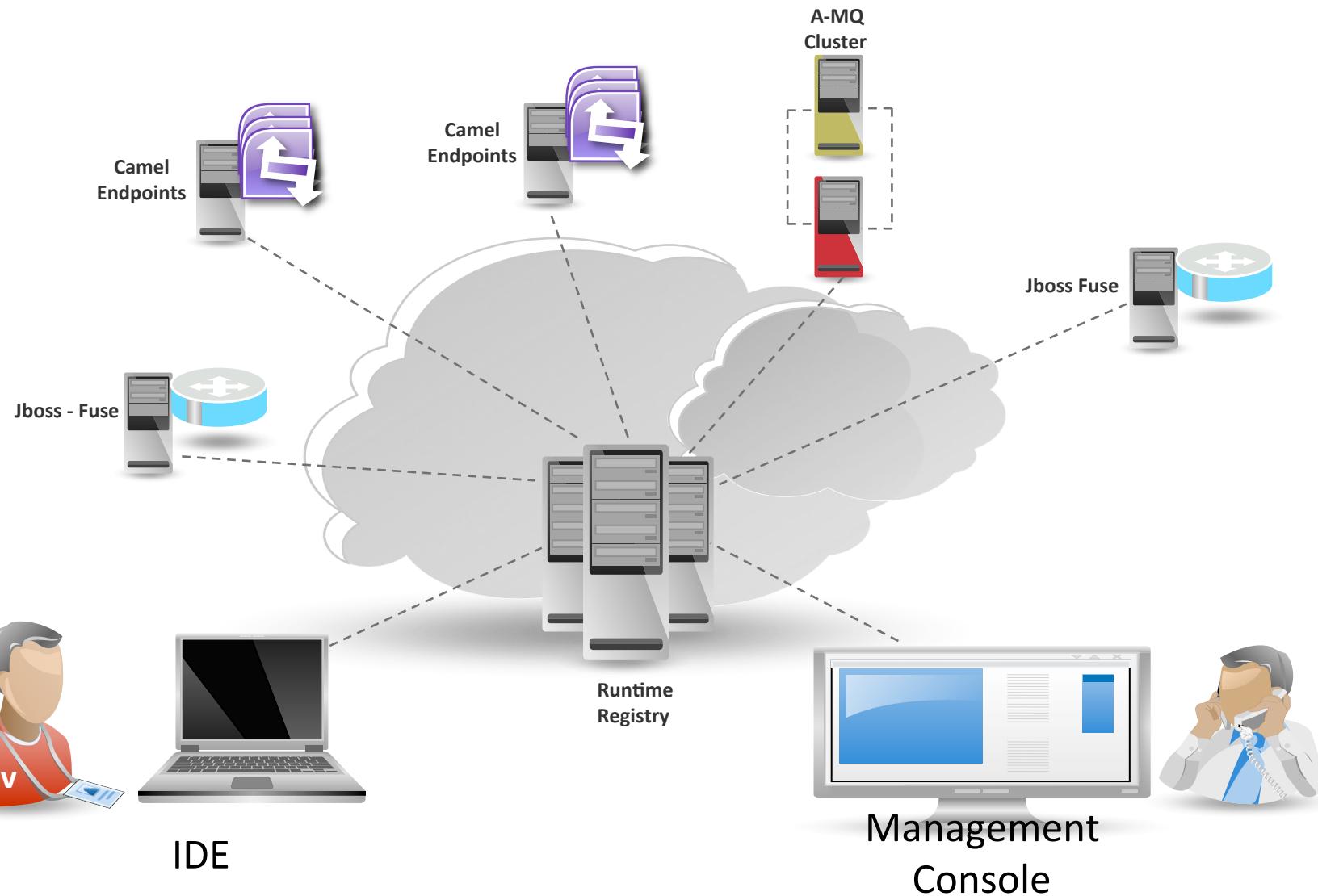
# JBoss Fuse – Key Projects



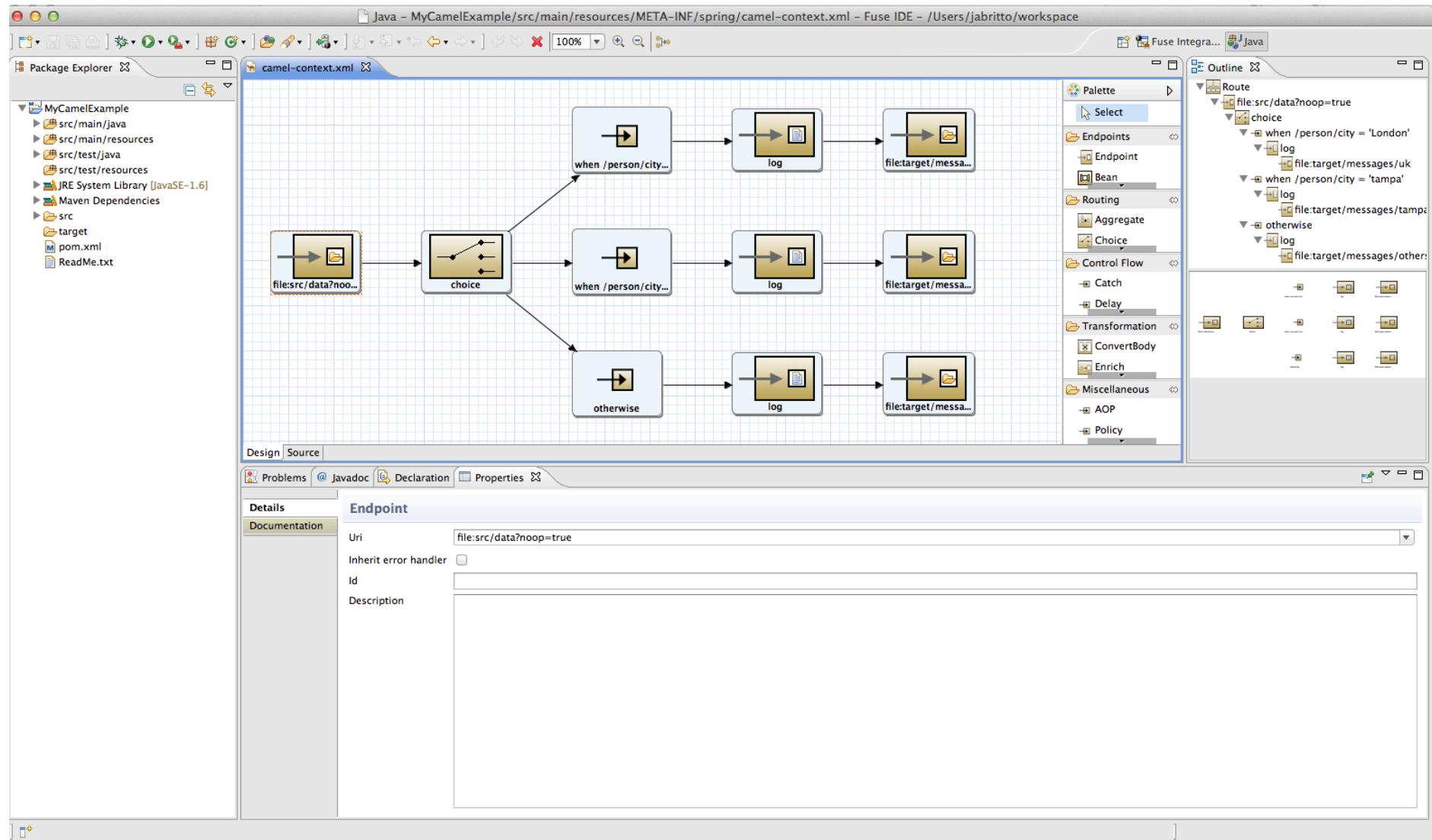
# JBoss Fuse - Karaf



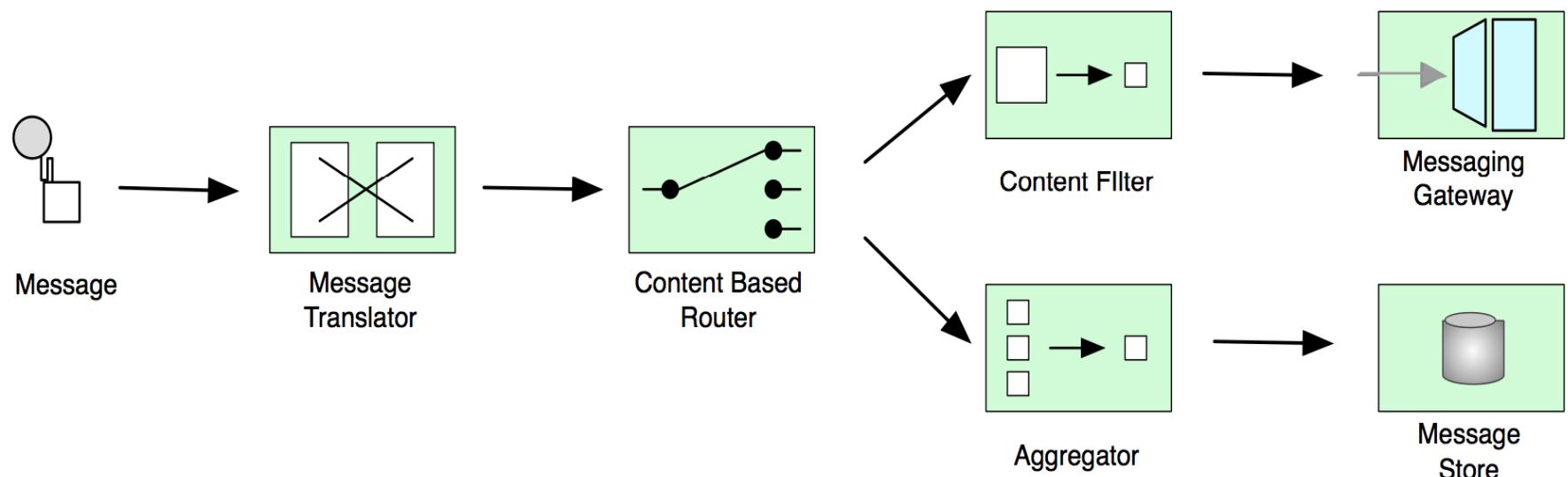
# JBoss Fuse - Fabric



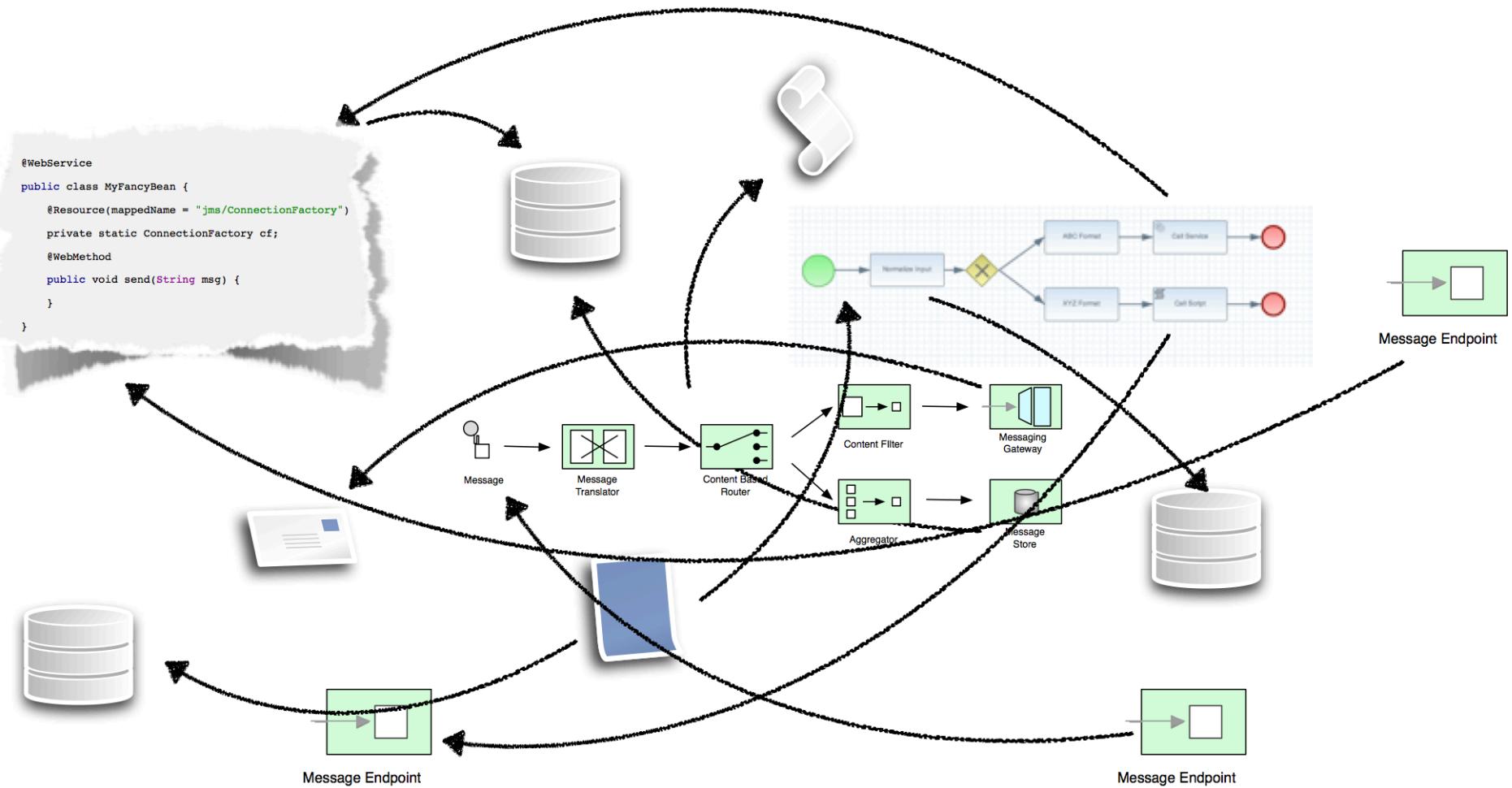
# JBoss Fuse – Visual Development



# Simplicity is good



# Reality isn't always simple



# JBoss SOA Platform

## Eclipse Visual Development

**Service  
Development**

**Core ESB**

**Design Governance**

**Runtime Governance**

## Operational Management

Core ESB and EAP

Structured, service-oriented development model

Service development as a first-order concept

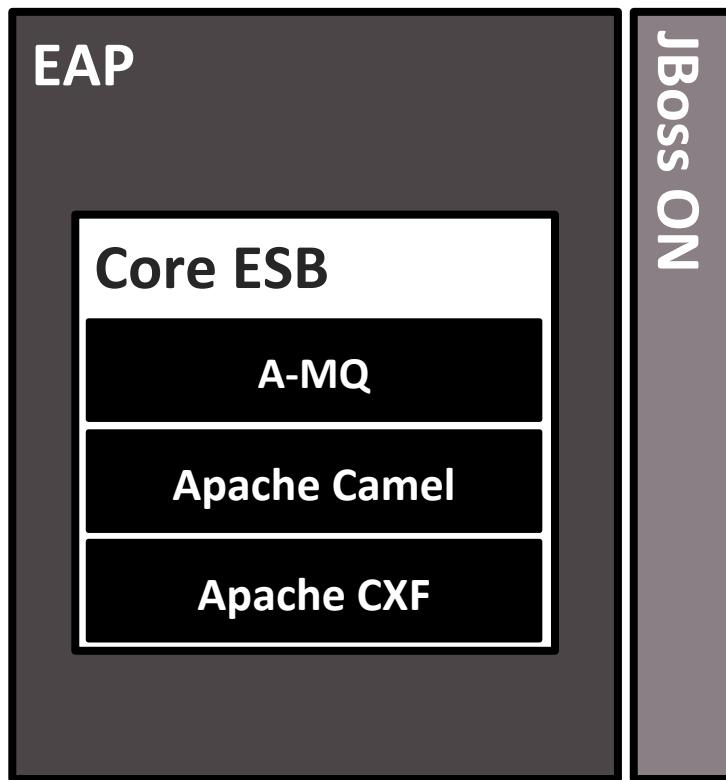
Code and model driven tooling

Simplified Service Lifecycle

Support for distributed, cross-functional teams

Functional equivalence with SOA 5

# JBoss SOA Platform



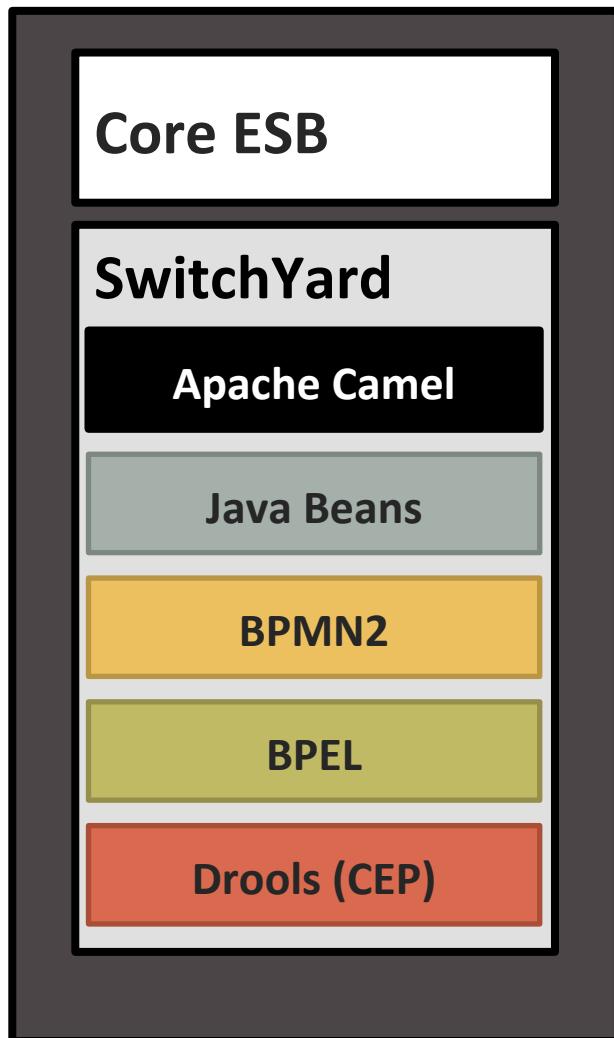
Core ESB running on EAP

Managed with base EAP capabilities or JBoss ON

Easy migration from Karaf to EAP

Choice of A/MQ or HornetQ

# Complex Services, Simplified



Switchyard Video Series

***Service First Development with Switchyard***

**Contract-based development** for messages, policy, SLA

**Dependency Management**

**Binding Abstractions** for interfaces & endpoints

**Declarative Transaction & Security Policy**

**Declarative Data Handling** for transformation & validation

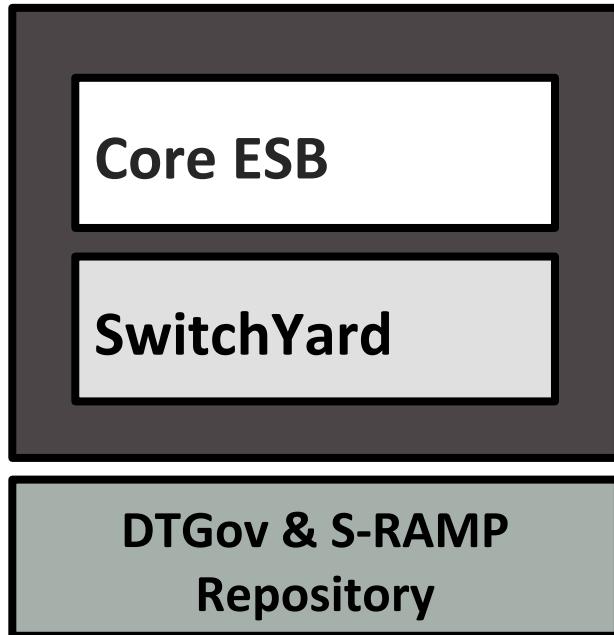
<http://vimeo.com/jbossdeveloper>

# Visual development

*Model focused visual development*

The image displays a composite screenshot of a visual development environment for application integration. At the top left, there's a graphical interface for defining routes and services, showing a flow from Service1 to a Process, which then connects to a Route and finally a Bean. To the right of this is a 'Bindings' palette listing various transport mechanisms. Below these are two separate component-based interfaces, each featuring a 'Component' node with a small coffee bean icon. The bottom half of the image shows two screenshots of the Eclipse IDE interface, specifically the 'Properties' view for security and transaction policies. The left screenshot shows the 'Security Policy' section with options for 'Client Authentication' and 'Confidentiality'. The right screenshot shows the 'Policy Details' section for a 'Transaction Policy', with 'None' selected and other options like 'propagatesTransaction' and 'suspendsTransaction' listed.

# Intelligent, Shared Repository



Service component and artifact registry based on Overlord DTGov & S-RAMP project

Store and publish services/policies across teams.

Notifications on content changes

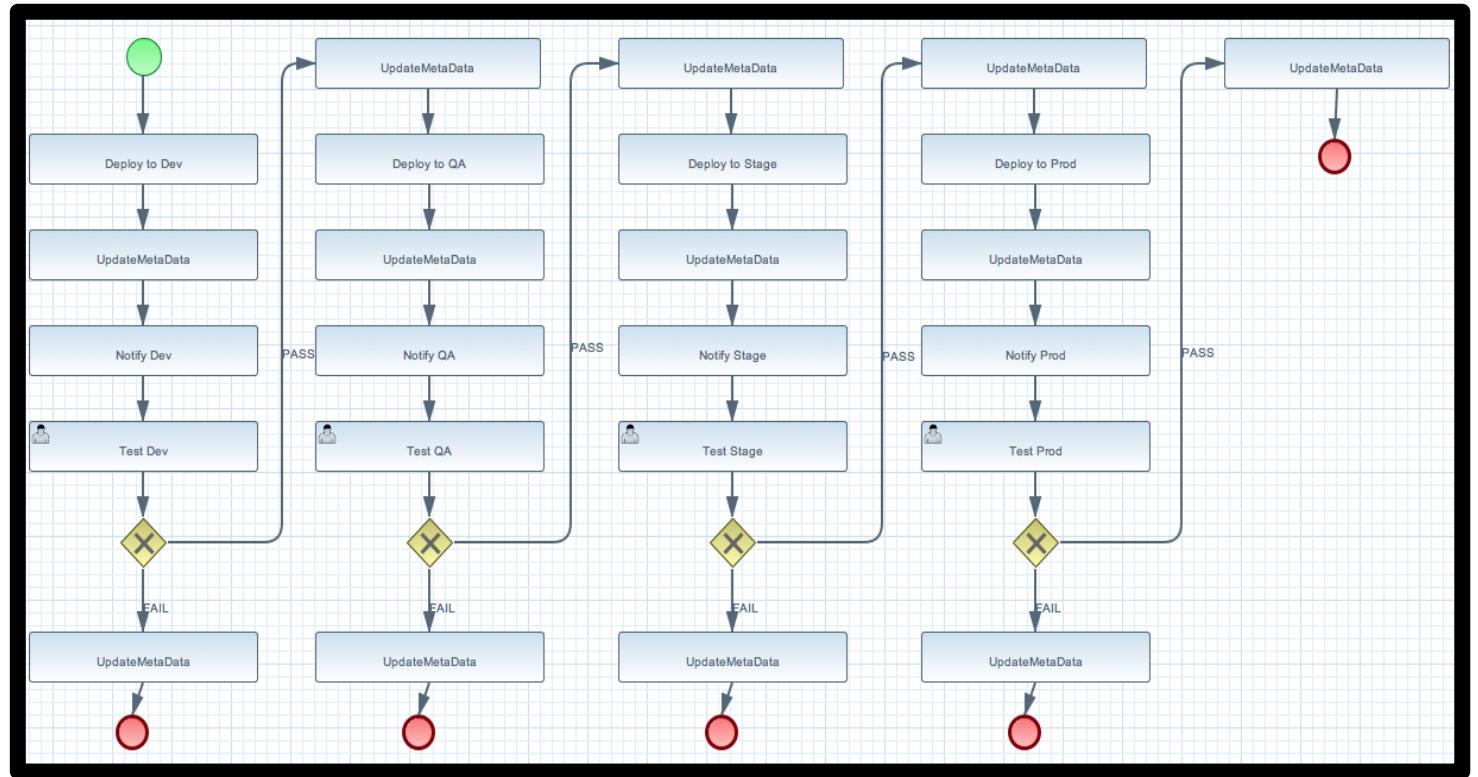
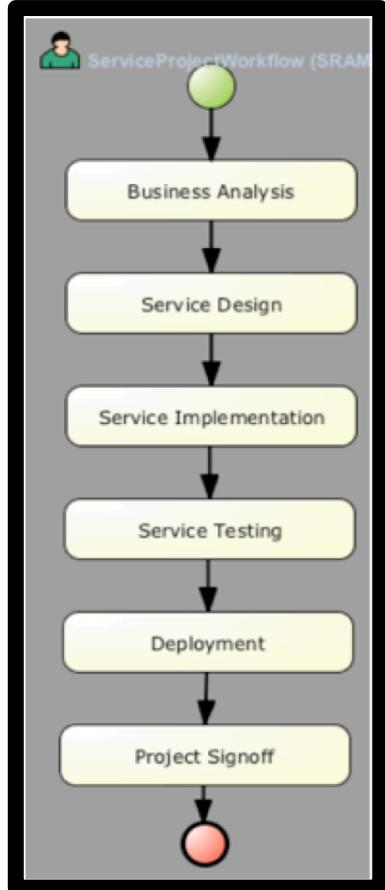
Track and Visualize artifact relationships

Impact Discovery & Analysis

Web and CLI tools to simplify search and reporting

# Lifecycle isn't just development

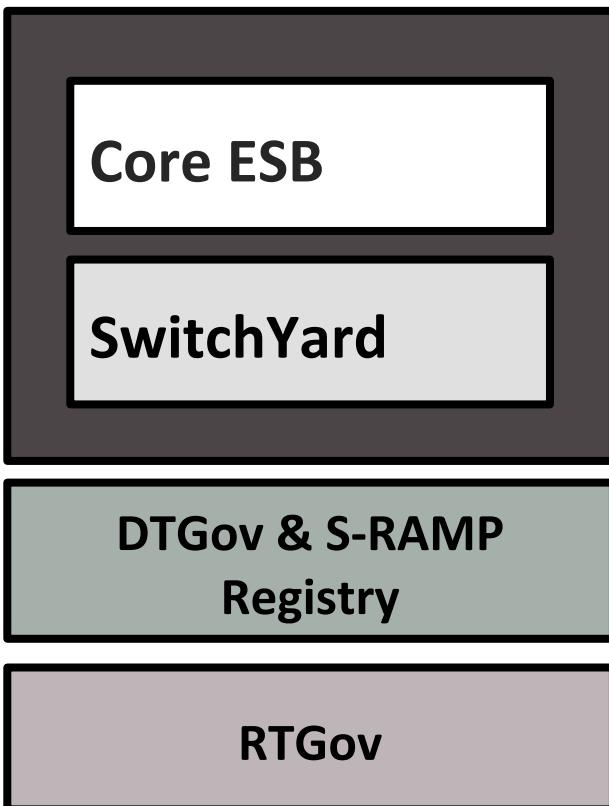
*Customizable, automated lifecycle workflow*



Workflow  
Repository (S-RAMP)

<http://bit.ly/11t5INz>  
<http://vimeo.com/50627742>

# Service Activity Monitoring



Overlord RTGov provides runtime governance

Real time activity reporting for service transactions

Event analysis and correlation within and across services

Runtime policy enforcement independent of service

Persisted information supports historic analysis of business activity

Results from the business policies can be cached for reference by other applications

# Visualize Services

*Customizable dashboard, comprehensive visibility*

The screenshot displays a customizable dashboard for monitoring services. On the left, there's a 'Response Time' chart showing peaks in response time over time. Below it is a 'Call Trace' section showing a sequence of service interactions. To the right, a 'Service Overview' section shows a network of services: OrderService, InventoryService, and Logistics Service, with their respective methods (makePayment, submitOrder, lookupItem, deliver) and performance metrics like count, average, min, and max.

Call trace details:

- submitOrder: [unswitched] - 28ms (100%)
- lookupItem: [unswitched] - 7ms (50%)
  - Information: Found the item. [3ms] (100%)
- deliver: [unswitched-q] - 7ms (50%)
  - Information: Delivering the item. [3ms] (100%)

Service Overview Metrics:

- OrderService**: makePayment (green), submitOrder (red)
- Inventory Service**: lookupItem (red)
- Logistics Service**: deliver (green)

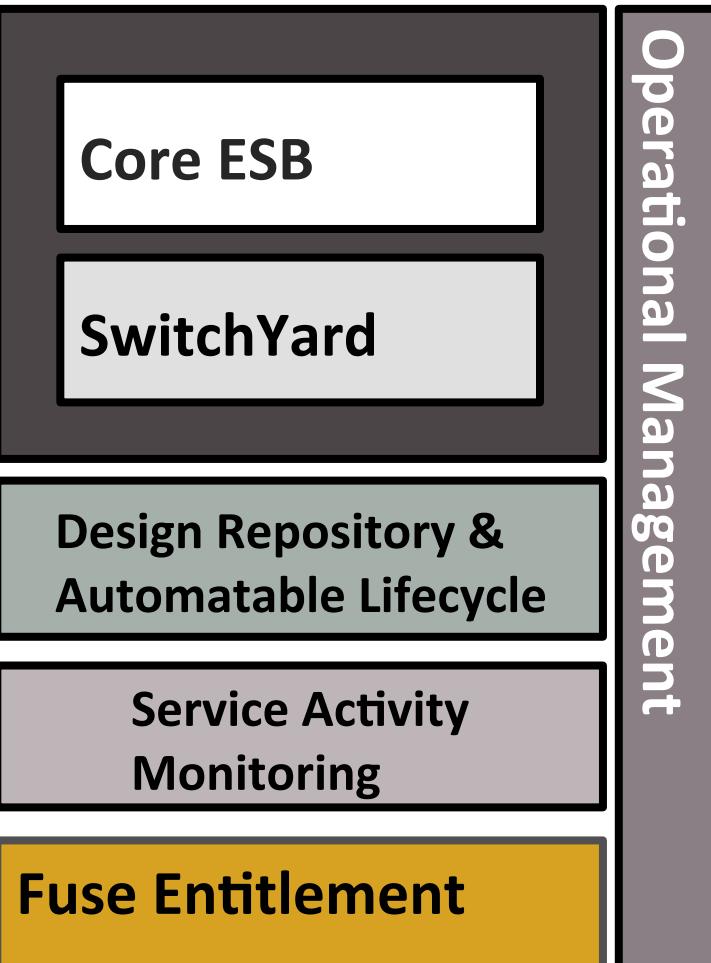
Type	Severity	Description	Updated Time	Time	Details
SLA Violation	Critical	OrderService exceeded maximum response time of 400 ms	2013-6-7 9:12:18	2013-6-7 9:12:14	
SLA Violation	Critical	InventoryService exceeded maximum response time of 400 ms		2013-6-7 9:12:14	

Call trace  
Activity Monitoring  
Define and Enforce a policy

<http://vimeo.com/54642223>  
<http://vimeo.com/51539146>  
<https://vimeo.com/54790975>

# JBoss SOA Platform v6

## Eclipse Visual Development



Builds on Fuse ESB and EAP

Structured, service-oriented development model

Services as a first-order concept

Simplified Service Lifecycle

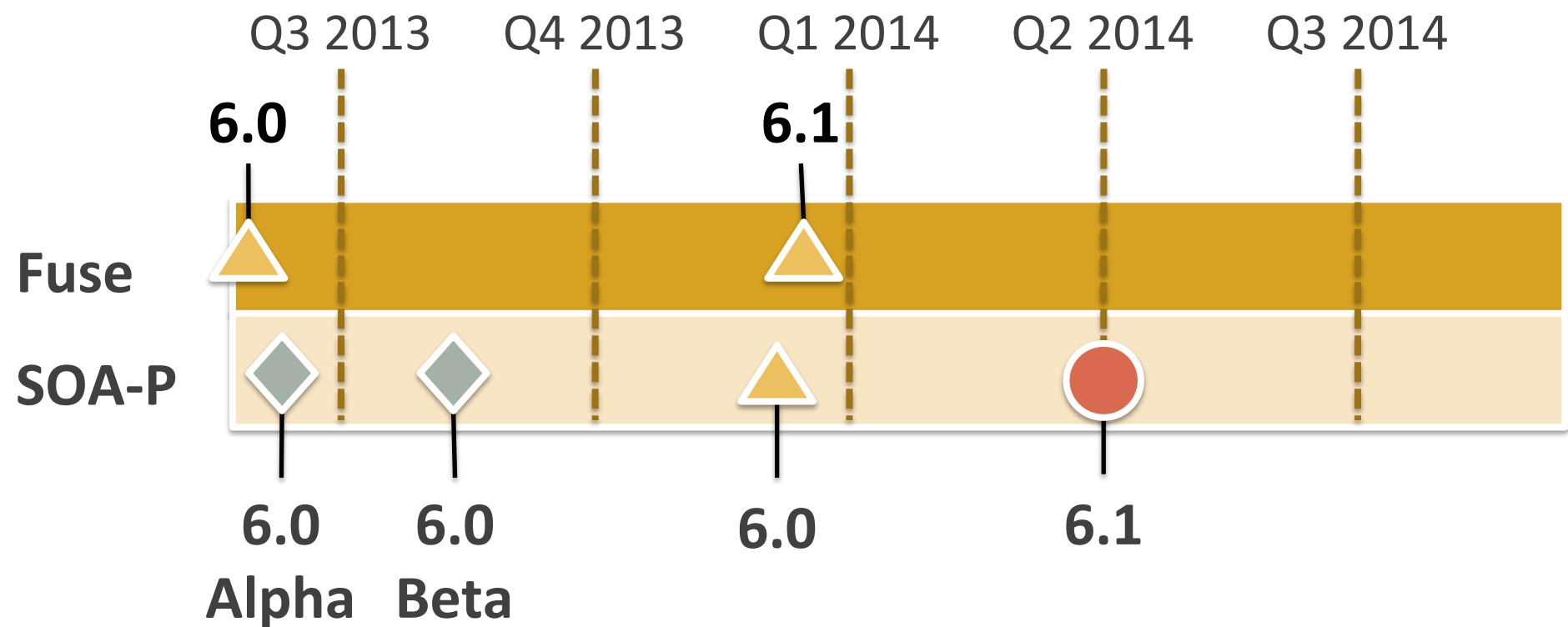
Visibility and Governance

Support for distributed, cross-functional teams

Code and model driven tooling

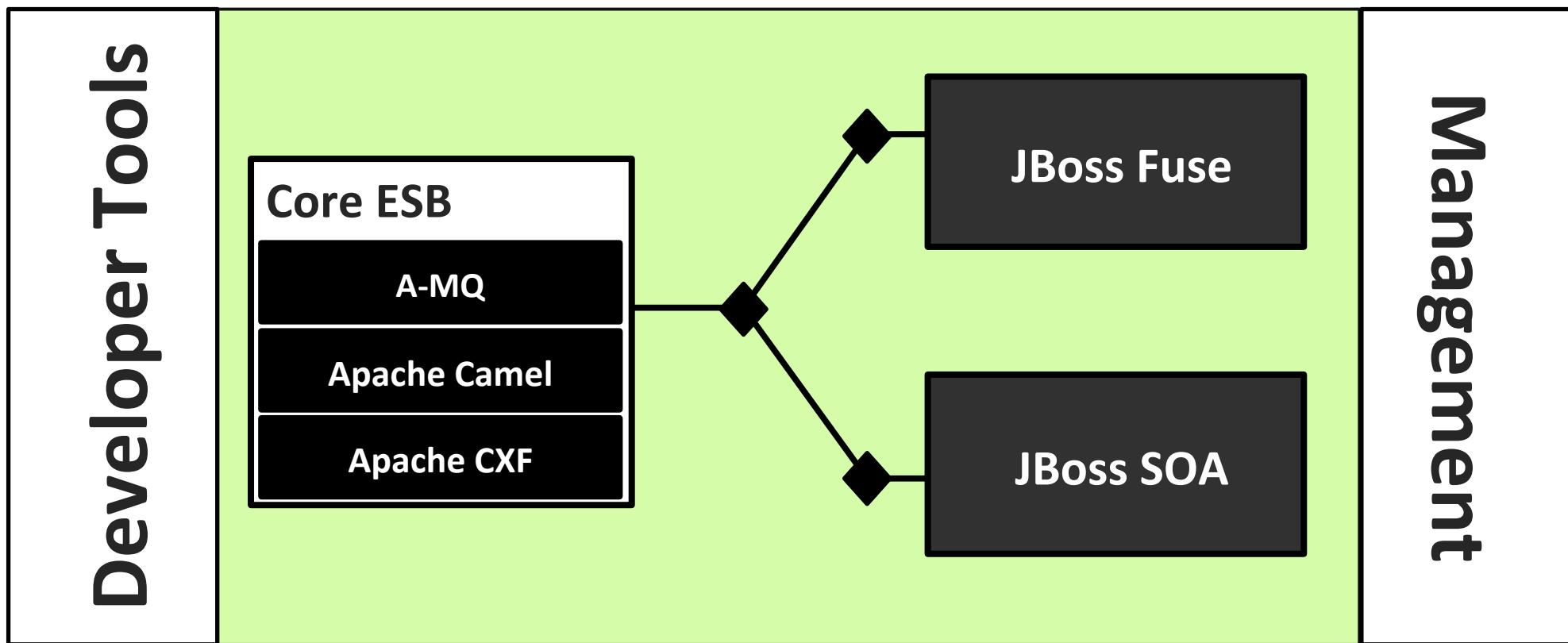
Functional equivalence with SOA 5

# Roadmap



SOA-P Early Access Program begins with Alpha, Open Beta with developer support

# Red Hat JBoss Fuse product line



# Other Sessions

## **Easy Integration with Red Hat JBoss Fuse**

Wednesday 3:40 – 4:40

Room 210

## **Red Hat Jboss SOA Platform 6 Quick Start**

Thursday 2:30 – 3:30

Room 210

## **Riding the Camel**

Wednesday 2:30 – 3:30

Room 206

## **Intro to SOA-6 Labs**

Wednesday 3:50 – 5:50

## **Choose your own Adventure**

Thursday 3:50 – 5:50

## **Migrating SOA 5 to 6 Labs**

Friday 9 – 11

# Q&A

**Q: I'm a current SOA-P 5 customer. What should I use moving forward?**

A: SOA-6. It is functionally equivalent to SOA-5, we will have migration tools to help move from 5 to 6 and will ensure interoperability between 5 and 6.

**Q: Will I still be supported with SOA-P v5?**

A: Yes, we have extended the lifecycle support for SOA-5 until 2018.

**Q: Can I move my projects from Fuse to SOA or vice-versa?**

A: Easily, depending upon features. e.g. If programming directly to Fabric or OSGI technology then you are currently limited to running in the Karaf container. There will be documentation and tooling to support migration of projects.

**Q: I want to use both SOA-P 6 and Fuse. How can I allocate my entitlements?**

A: Fuse entitlements are included with SOA-P 6 subscriptions. Core counts can be split between deployments