

Next Generation Open Source

SOA

Burr Sutter

Sr. Product Manager, SOA
(JBossESB, Riftsaw, jBPM, Drools)

September 3, 2009



Agenda

Is SOA Dead?

JBoss SOA Platform Overview & Roadmap

RESTful inclinations

Complex Event Processing (CEP) w/ Drools Fusion

Infinispan + ESB

BPEL

Is SOA Dead?

**"The report of my death was
an exaggeration."**

Mark Twain



SOA, ROA & WOA – Oh My!

ROA – Resource Oriented Architecture - REST (Representational State Transfer – Roy Fielding)

WOA – “Web Oriented Architecture (WOA) is a style of software architecture that extends service-oriented architecture (SOA) to web based applications, and is sometimes considered to be a light-weight version of SOA. WOA is also aimed at maximizing the browser and server interactions by use of technologies such as REST and POX.” - Wikipedia

SOA is NOT SOAP

- architectural style vs wire protocol

SOA is NOT WS-*

- patterns & principles vs a large body of standards

SOAP vs REST Technical Differences

HTTP Verbs:

SOAP – POST

REST – GET, PUT, POST, DELETE

Contract Definition:

SOAP – WSDL – operations & messages

REST – HTTP Verbs are the operations, messages may or may not have a contract via XSD and/or JSON (POX – plain 'ol XML)

Content-Type:

SOAP – focused on XML

REST – allows for any payload – XML, JSON, Atom, RSS, etc.

Clients:

SOAP – requires a client that understands WSDL

REST – requires a client that understand HTTP

Forget Services, I want Events! - EDA

EDA – “*Event Driven Architecture is a software architecture pattern promoting the production, detection, consumption of, and reaction to events. An event can be defined as a significant change in state*” - Wikipedia

Services have historically been more request-response focused – synchronous processing – normally for responding to awaiting users/consumers.

Events are more real-time alerting, no waiting – asynchronous processing – think push.

“*Event-driven architecture can complement service-oriented architecture (SOA) because services can be activated by triggers fired on incoming events*” - Wikipedia.

JBoss SOA Middleware

Overview

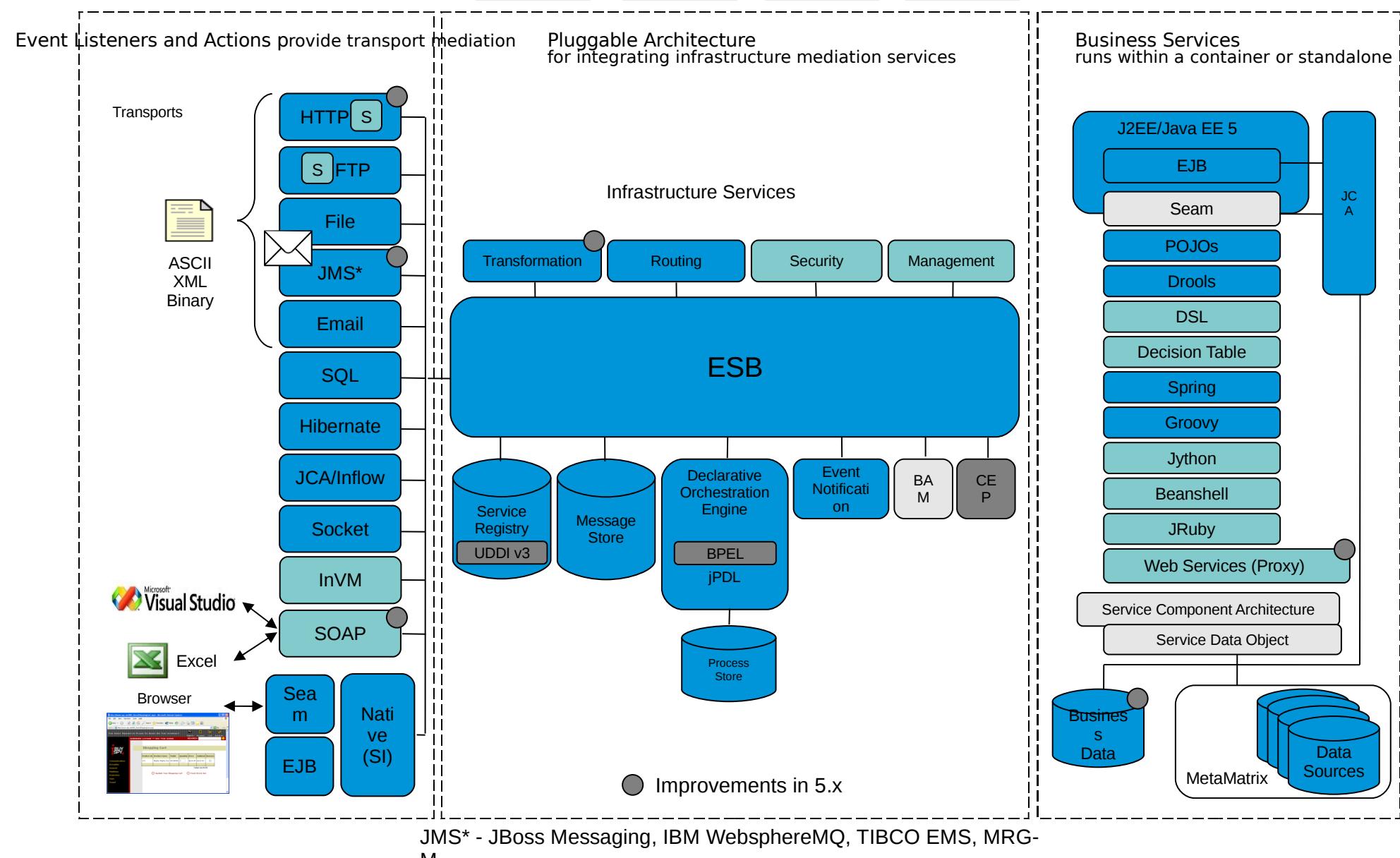


4.2

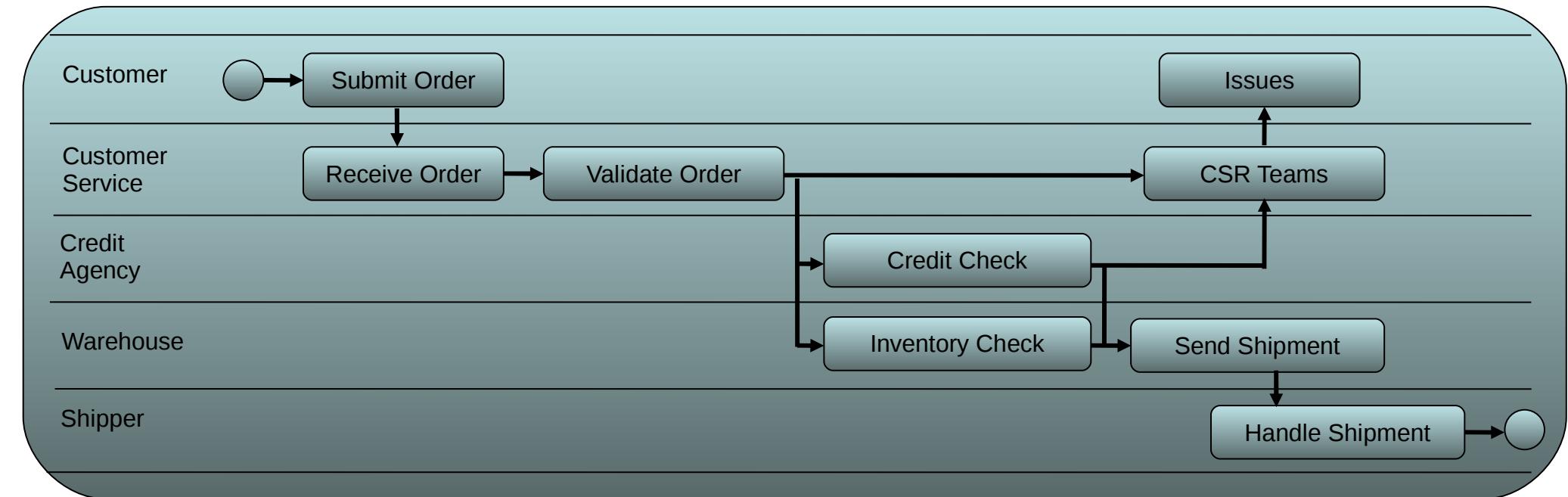
4.3

5.0

Future



Beyond ESB: Context



Validate Order

- a Parse XML
- b Transform
- c Apply Business Rules

Credit Check

- a Create Outbound Msg
- b Handle Response
- c Apply Business Rules

Send Shipment

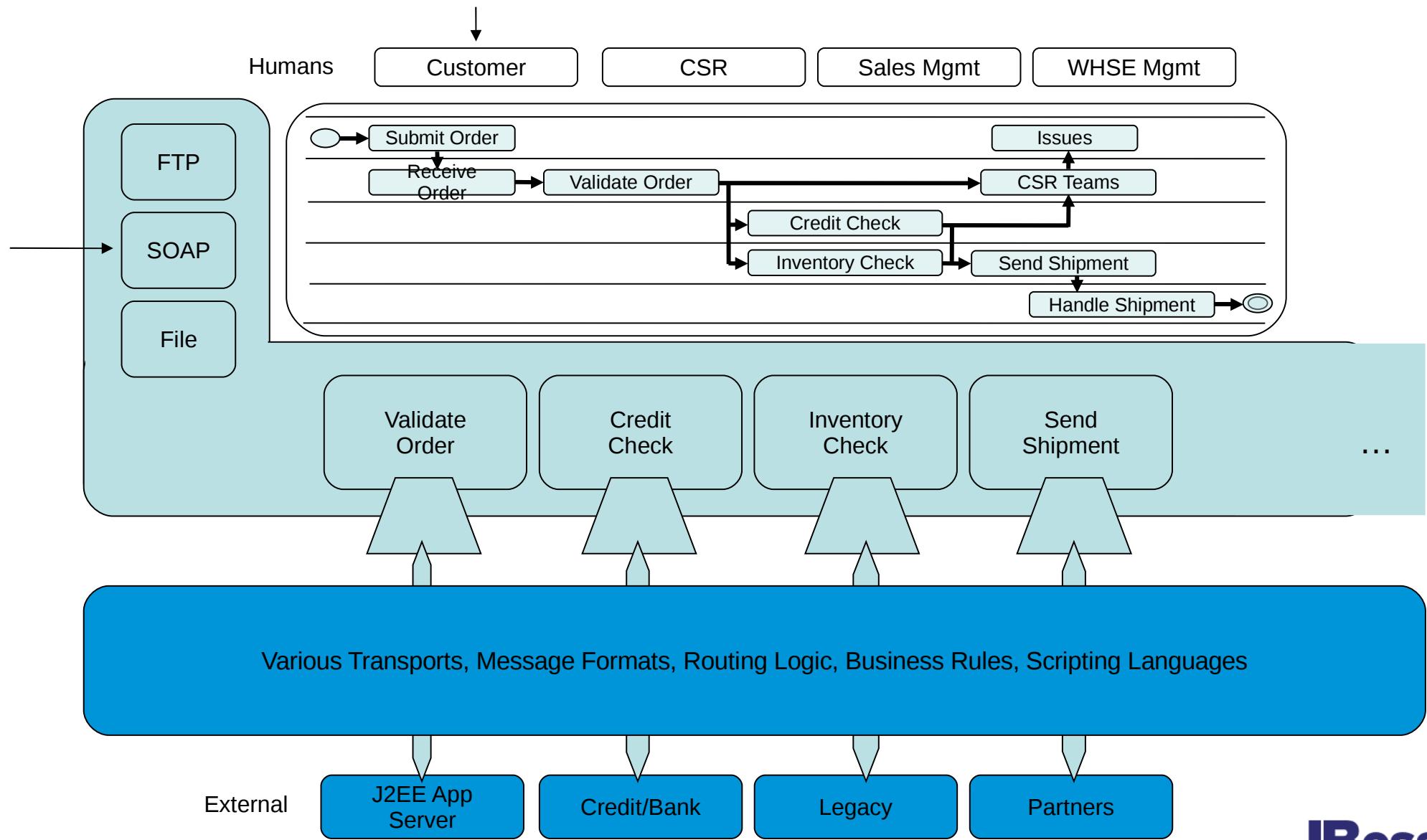
- a Determine Shipper(s)
- b Print Labels
- c Print Pick Tickets
- d Create & Send ASNs

ESB Mediates & Provides Services

Inventory Check

- a Send to N Warehouses
- b Handle N Responses
- c Determine Best WHSEs
- d Handle Drop-Ships

Synergy of ESB + BPM + Rules



SOA Monitoring and Management

The image displays four screenshots of JBoss SOA monitoring tools:

- JON Monitor Resource Indicators - JBoss ESB Services - Mozilla Firefox**: Shows a list of resources including Central_Operations_Service, Credit_Check_Service, East_Operations_Service, Inventory_Check_Service, Receive_Order_Service, Shipping_Service_Service, Starter_Service_Service, Validate_Order_Service, and West_Operations_Service. Each resource has a status icon (green, yellow, red) and an 'Action' button.
- JON Metric Chart - Mozilla Firefox**: A line chart titled "Overall Service Message Count" showing data from 9:10 to 12:52. The Y-axis ranges from 0 to 1,925.2. The chart shows a sharp increase from 9:10 to 10:30, peaking at 1,835, and then remaining relatively stable around 1,925.2. A legend indicates "Actual" (blue line), "Average" (red line), and "Baseline" (green line).
- JON Dashboard - Mozilla Firefox**: A central dashboard with various monitoring panels. It includes sections for "Auto-Discovery" (no resources), "Recently Added Resources" (localhost), "Favorite Resources" (Receive_Order_Service, Shipping_Service_Service), and "Recent Alerts" (no alerts). Summary counts for Platform, Server, and Service are provided.
- JON Monitor Resource Indicators - localhost - Mozilla Firefox**: A detailed view for the localhost resource. It shows basic information (Type: Windows (Platform), Version: Win32 6.0, Hostname: Burr-PC, OS Version: 6.0, Architecture: x86), a monitor tab with resource status, and a chart for "Free Memory" (localhost) showing usage from 9:10 to 12:52. The chart shows a peak of 974.5MB at 10:30 and a low of 817.2MB at 9:10.

REST



Value of REST with an ESB

JBossESB is not hard-wired into Web Services nor JMS, it is async event-driven and also allows for sync request-response.

RESTful style interactions (GET, PUT, POST, DELETE) allow for Groovy, JavaScript, Ruby, etc clients to more easily consume a service.

JBossESB now allows for different content types and use of HTTP response codes.

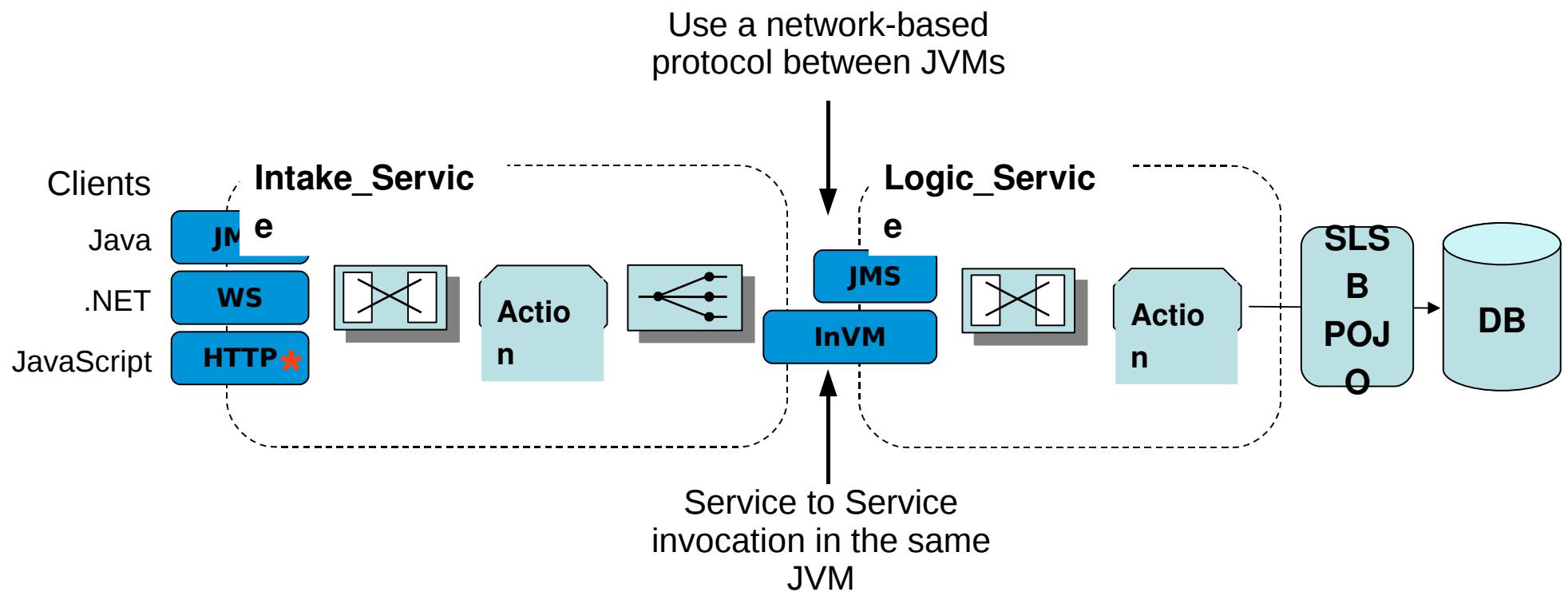
REST is based on HTTP – new http-provider (ESB 4.7)

New HTTP Provider (jboss-esb.xml)

```
<providers>
  <http-provider name="http">
    <http-bus busid="http_gateway"/>
    <exception
      mappingsFile="/http-exception-
mappings.properties" />
  </http-provider>
</providers>

<services>
  <service category="Sales" name="List"
    description="" invmScope="GLOBAL">
    <listeners>
      <http-gateway name="sales"
        busidref="http_gateway"
        urlPattern="sales/*" />
    </listeners>
    <actions mep="RequestResponse">
      <action name="createAtomFeed"
        class="atom_publisher.MyAction"/>
    </actions>
  </service>
</services>
http://localhost:8080/MyESBArchive/http/sales/*
```

ESB Transport Mediation

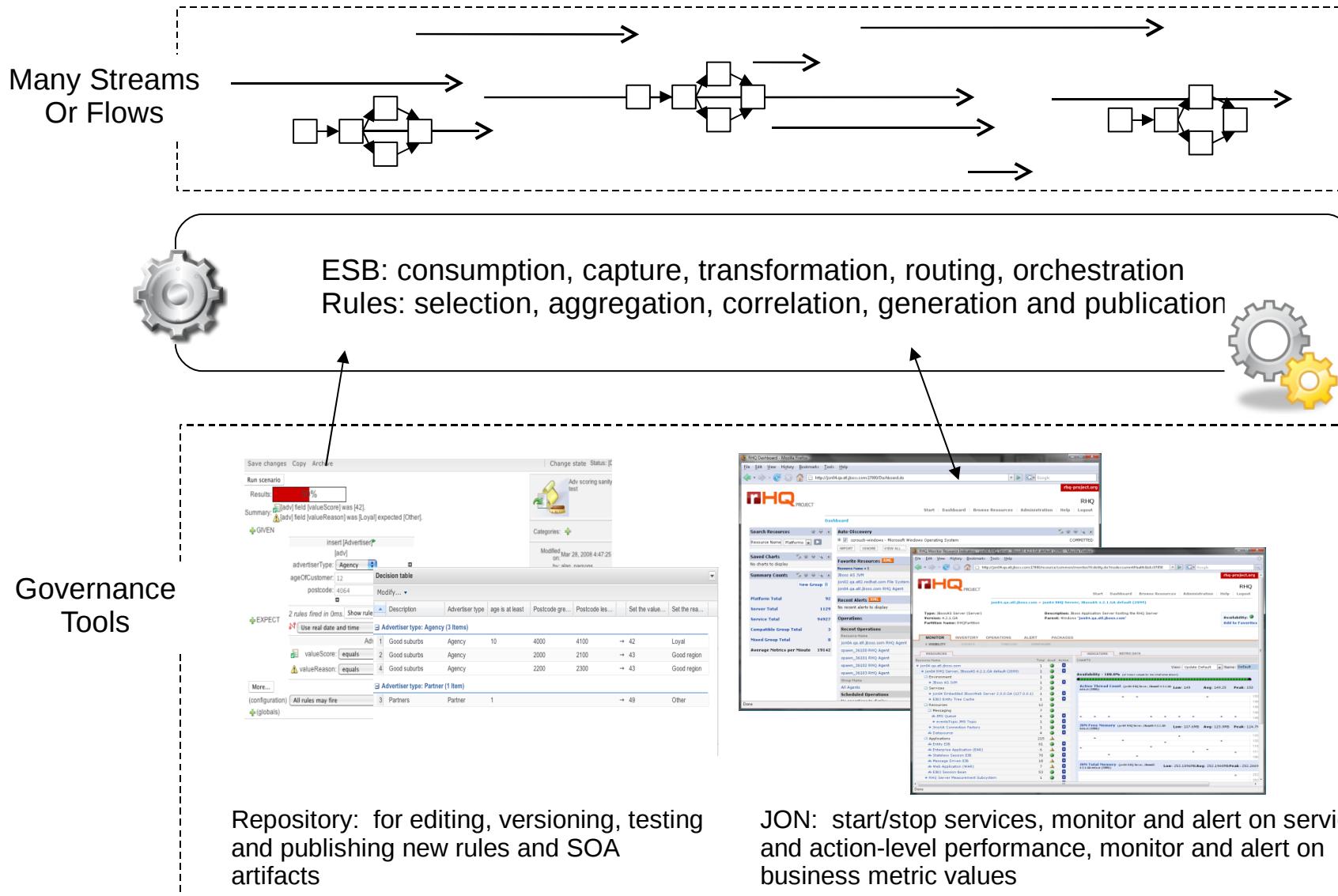


ATOM Demo

Complex Event Processing



ESP + CEP via ESB, Rules & JON

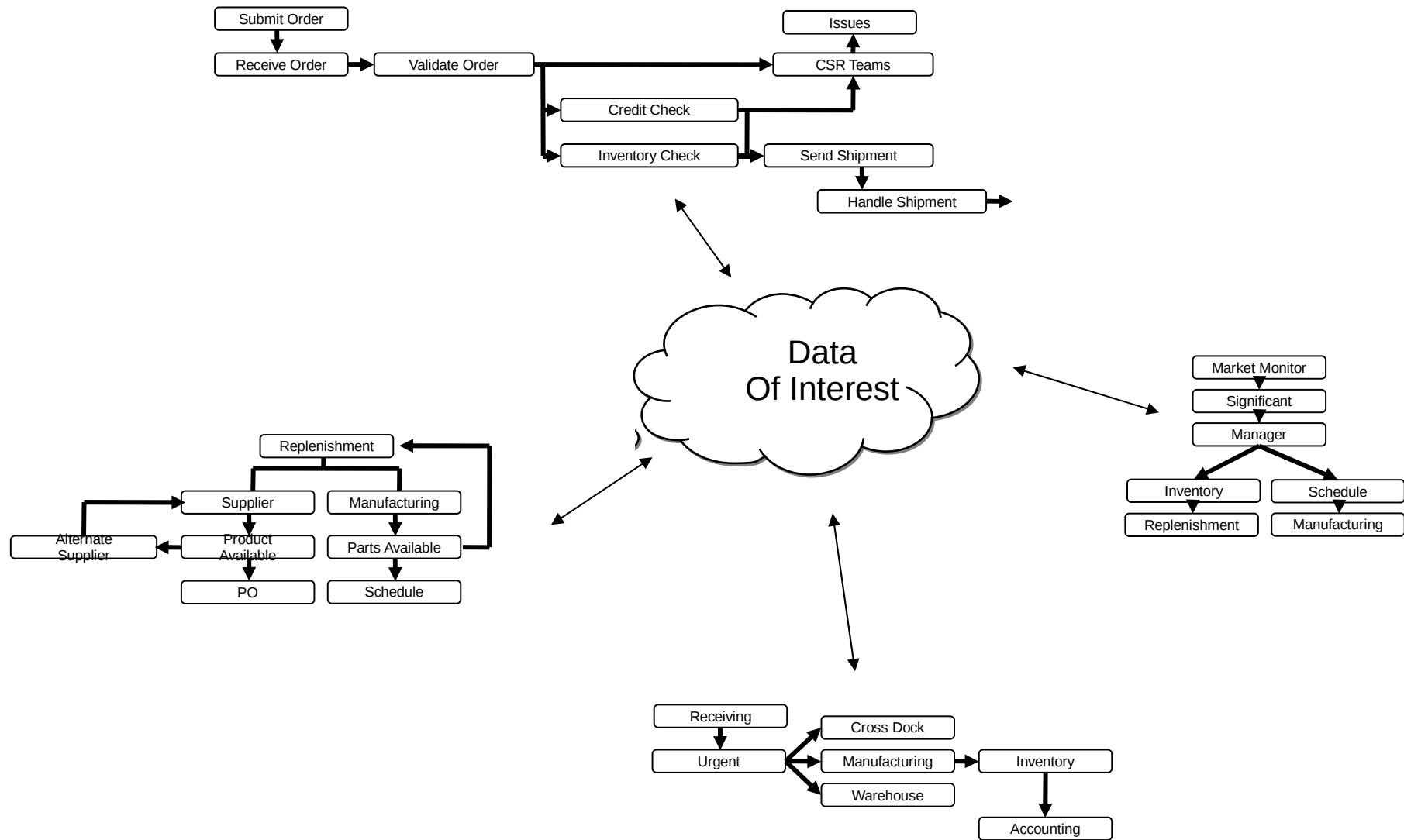


Event Declaration & Selection

```
declare StockTick
  @role( event )
  @expires( 2m )
end

rule "average over last minute"
when
  $stat : Statistics( $symbol : symbol )
  Number( $av : doubleValue ) from accumulate(
    StockTick( symbol == $symbol, $p : price )
  over window:time( 1m )
    average( $p ) )
  from entry-point
then
  modify( $stat ) {
    average = $av
  }
end
```

Deal flow – Business Interceptor



Value of CEP – Drools Fusion

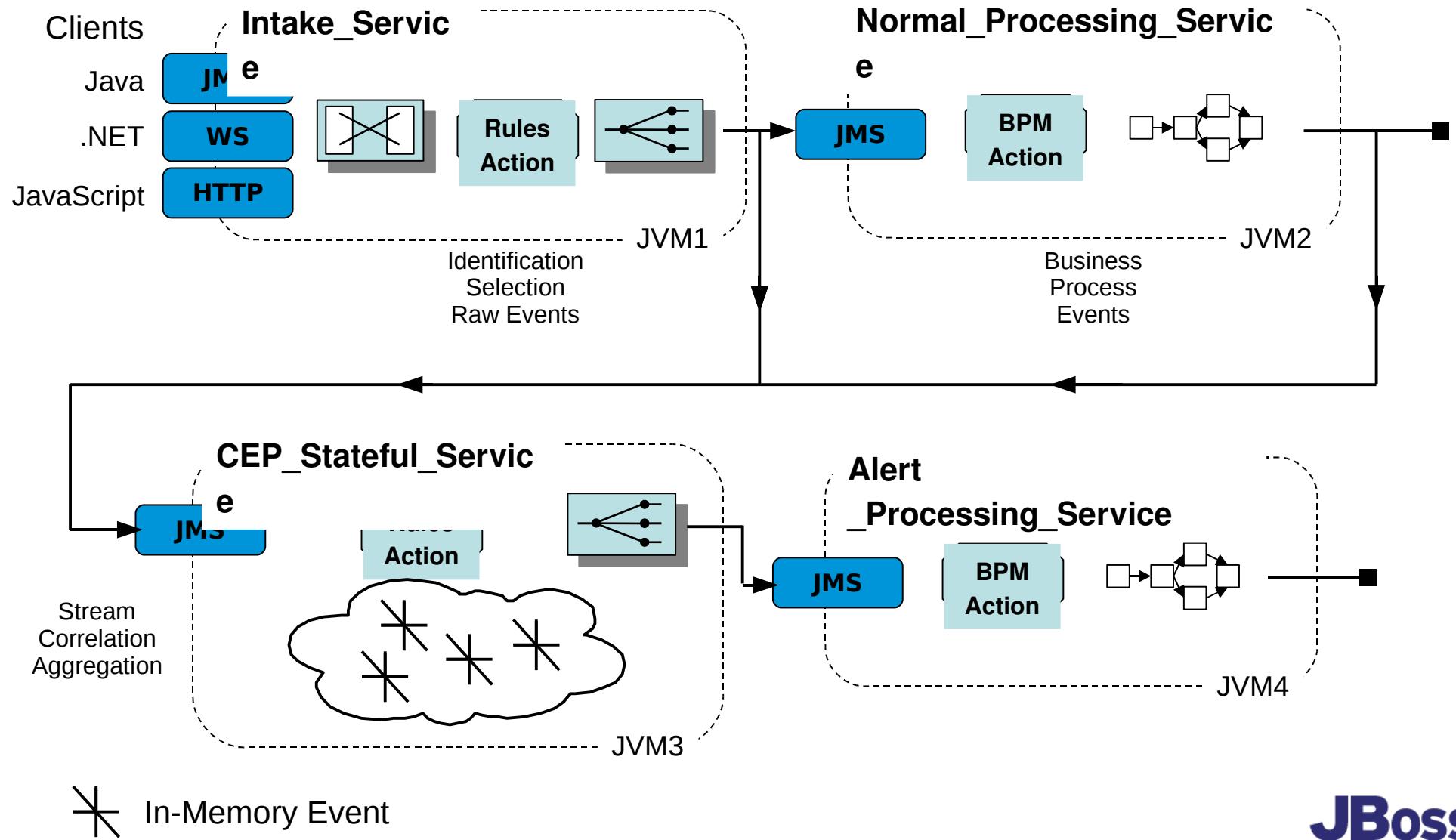
Stateful Rules Engines live in ESB Nodes, they retain the real-time history of the event flow

Rules look for significant/interesting events in the stream, expiring older, insignificant events (memory management) – stateless rules engines/ESB nodes can route to stateful engines.

Event data can be aggregated/accumulated over a time window or event series

New events or messages back through the ESB can be generated, providing an alerting mechanism.

CEP via JBossESB



CEP Demo

Infinispan + ESB



Quick Introduction: Infinispan

Spiritual successor to JBoss Cache > Data Grid

Still a “peer to peer” architecture, client-server is also available

JSR 107 – JCACHE

Speaks REST, memcached and a custom binary protocol

Improve response times for service requests

Reduce load on critical backend systems

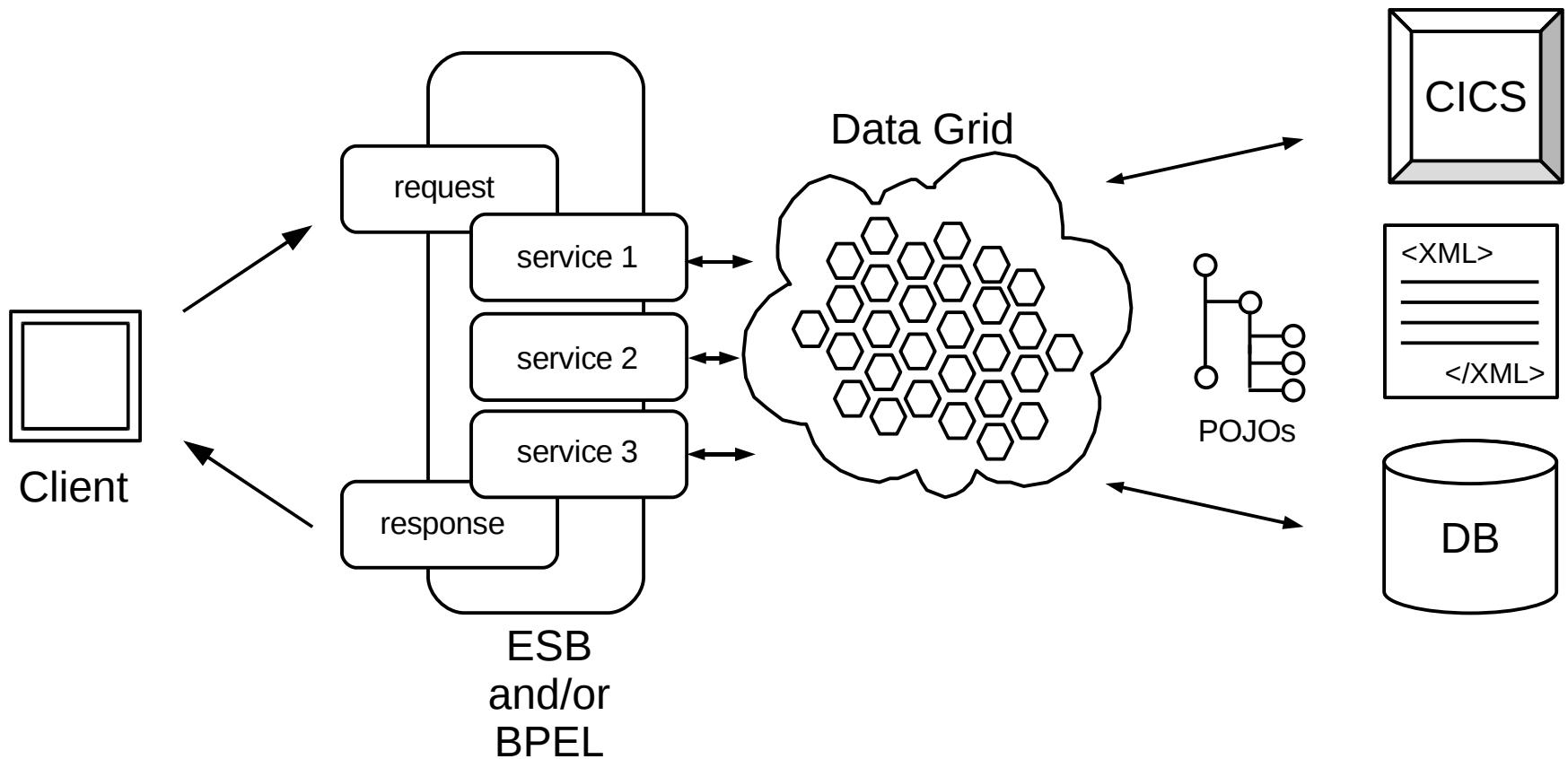
Improve fault tolerance, throughput, performance

Linear scalability

Predictable latency under increasing load

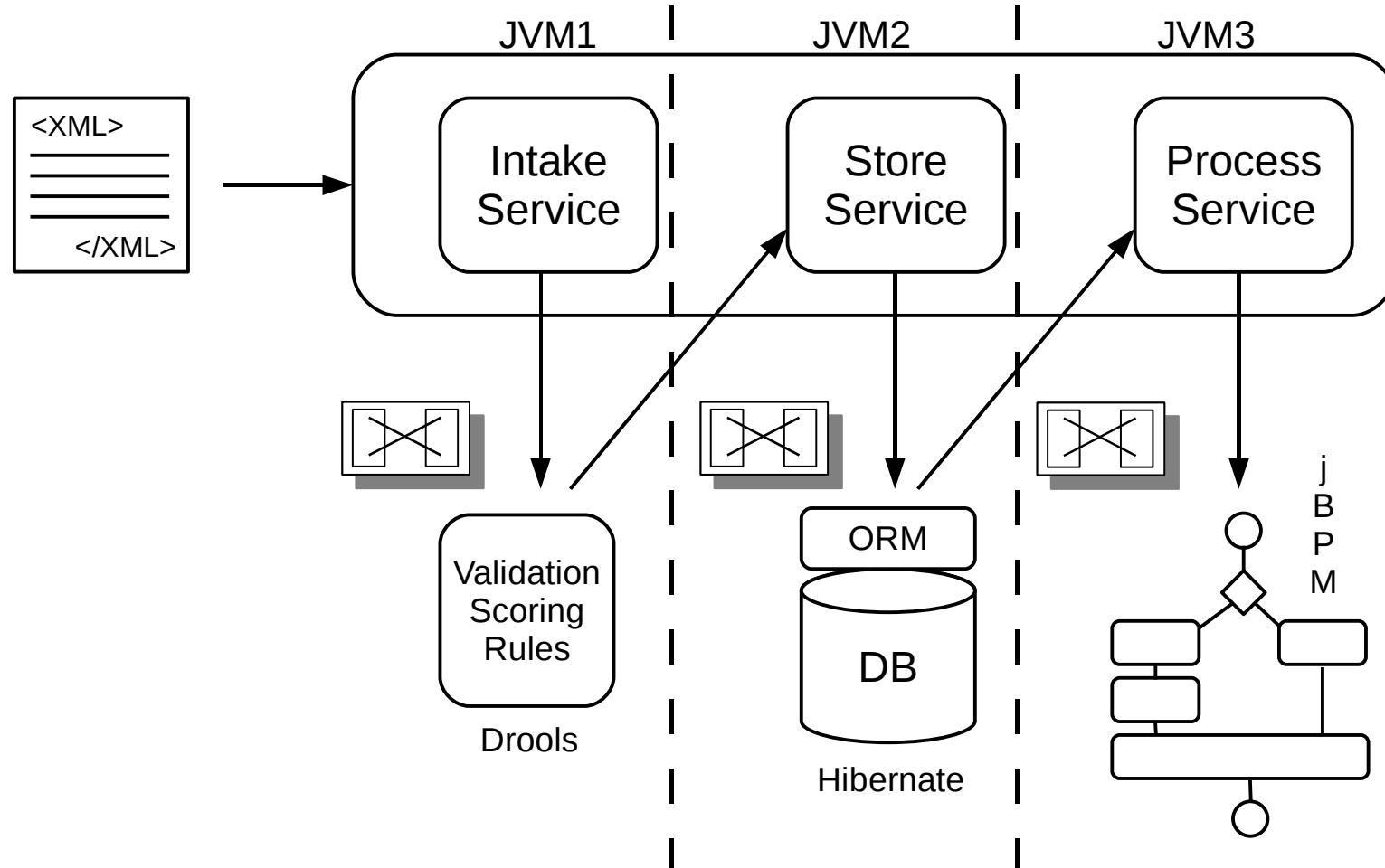
Access to more than a single JVM's Heap

Service Result Cache



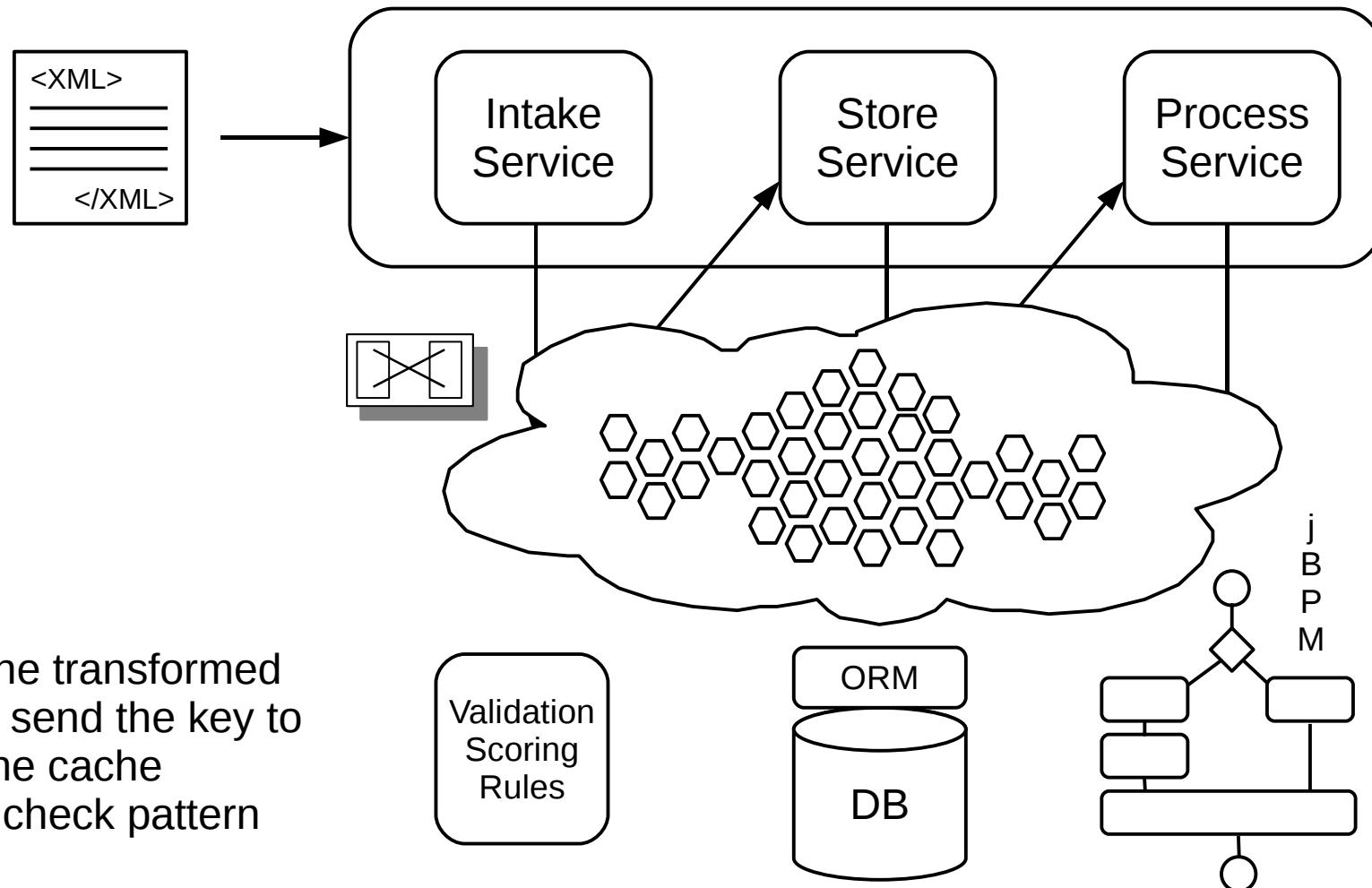
Results of service invocations to “expensive” resources
can be cached by the middle-tier architecture.
Transformations to POJOs can be time consuming

Hoppity, Hop, Hop



Serializable large XML, transforming multiple times,
serializing large POJOs across multiple network/JVM boundaries
can be expensive

Still Hopping, less load



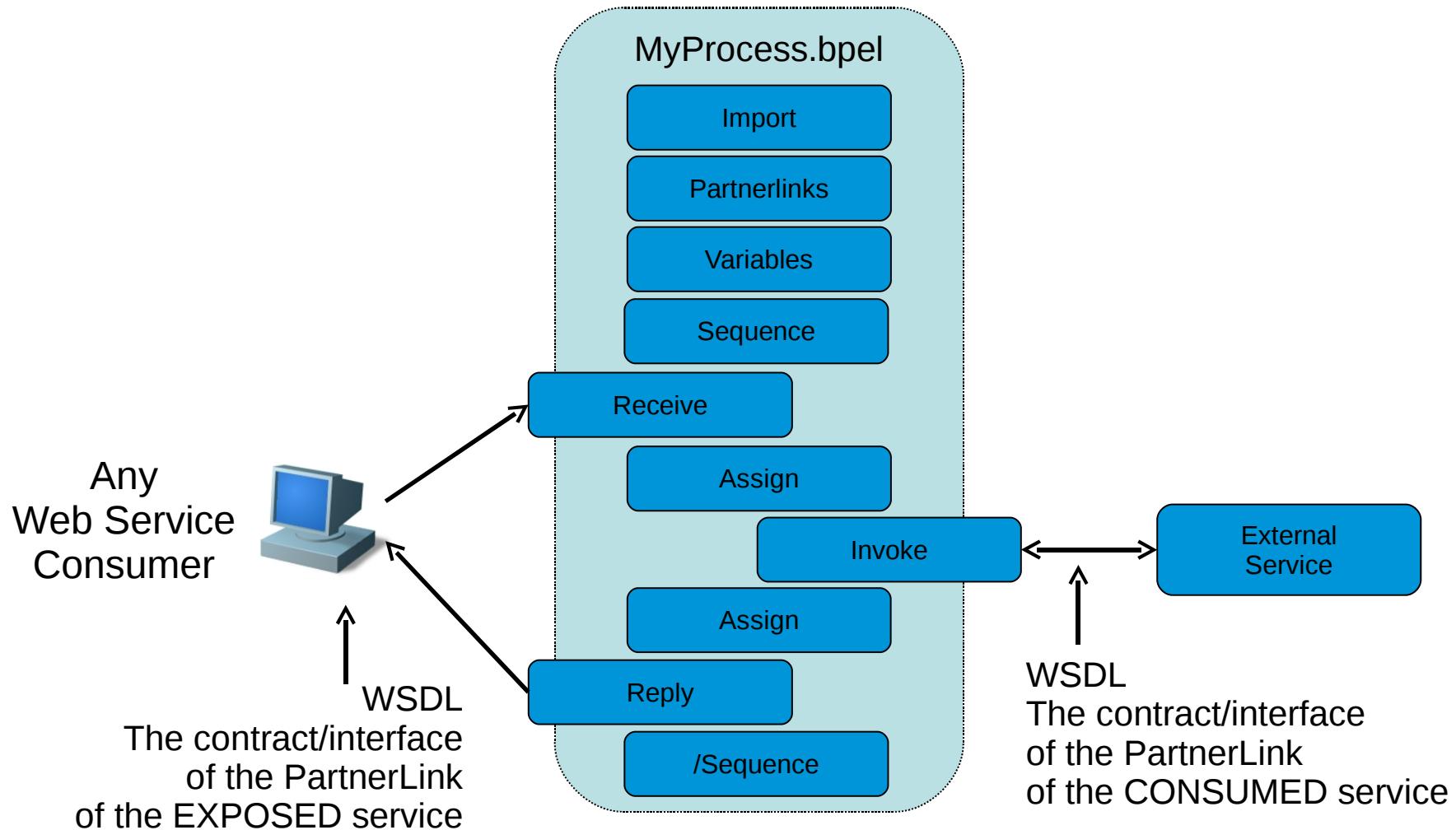
Data Grid Demo

JBoss
WORLD
CHICAGO 2009

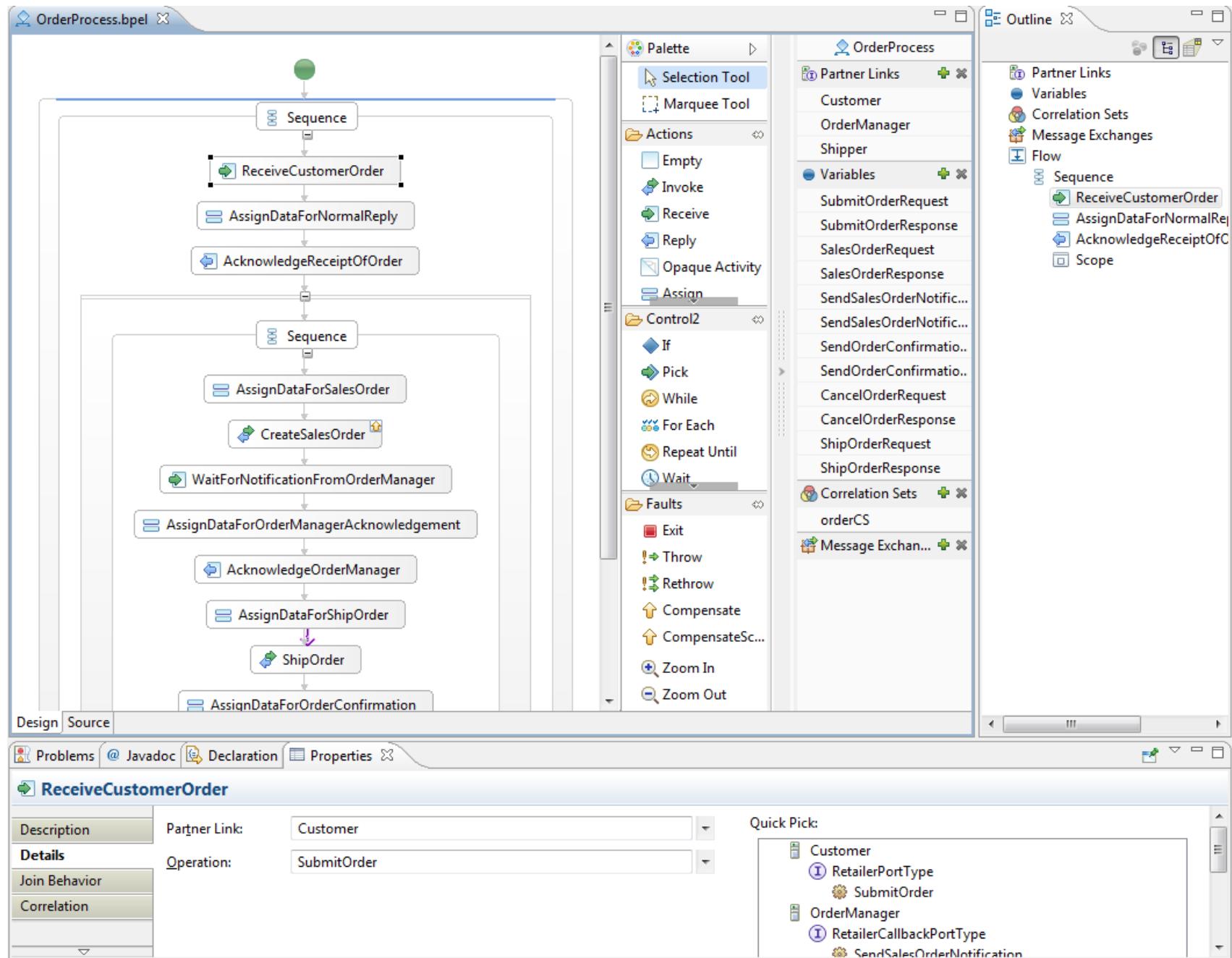
BPEL



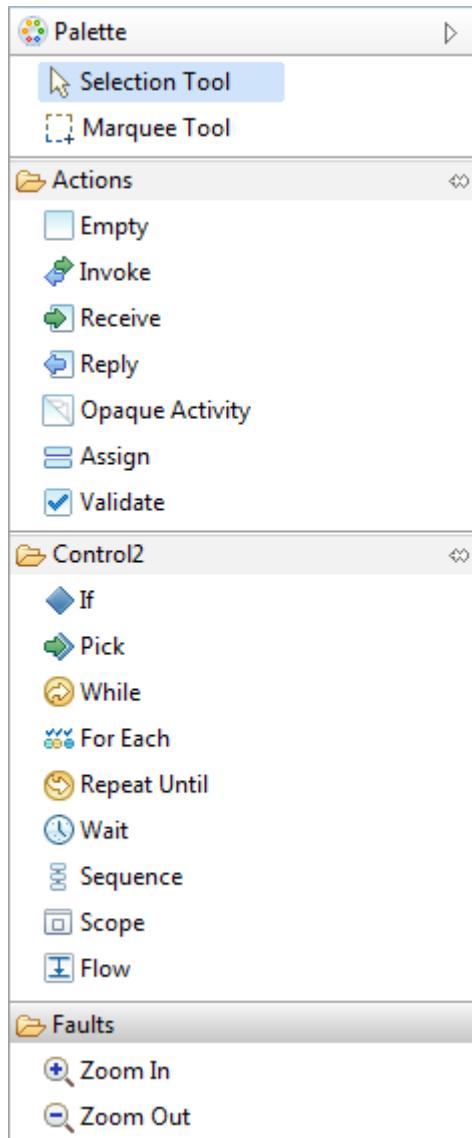
Simple BPEL Process Structure



Tools – BPEL Editor



BPEL Palette



Partner Link Properties

The screenshot shows the JBoss Integration Studio interface with the 'OrderProcess.bpel' project open. The left sidebar lists various components: Partner Links (Customer, OrderManager, Shipper), Variables (SubmitOrderRequest, SubmitOrderResponse, SalesOrderRequest, SalesOrderResponse, SendSalesOrderNotificationRequest, SendSalesOrderNotificationResponse, SendOrderConfirmationRequest, SendOrderConfirmationResponse, CancelOrderRequest, CancelOrderResponse, ShipOrderRequest, ShipOrderResponse), Correlation Sets (orderCS), and Message Exchanges (Message Exchanges).

The main area displays the 'Properties' view for a 'Customer' partner link. The 'Description' tab shows 'Partner Link Type: PurchasingPLT' and a 'My Role' section where 'Seller' is selected. The 'Partner Role' section shows 'Buyer' selected. The 'My Operations' section lists 'SubmitOrder', 'SubmitOrderRequest' (with sub-items 'Document: customerOrder' containing 'header: OrderHeader' and 'items: OrderItems'), and 'SubmitOrderResponse' (with sub-item 'Document: customerOrderAck'). The 'Partner Operations' section lists 'SendOrderConfirmation', 'SendOrderConfirmationRequest' (with sub-item 'Document: orderConfirmation' containing 'customerNumber: string', 'poNumber: string', and 'orderNumber: string'), and 'SendOrderConfirmationResponse' (with sub-item 'Document: orderConfirmationAck').

Tools – WSDL Editor

The screenshot shows the JBoss Seam WSDL Editor interface. The top navigation bar includes tabs for OrderProcess.bpel, BPELRetailer.wsdl, Customer.wsdl, and OrderManager.wsdl. The OrderManager.wsdl tab is active, displaying a service interface and its operations.

OrderManagerPortTypeService (Port Type):
OrderManagerPortTypePort (Port):
http://localhost:8865

RetailerCallbackService (Port Type):
RetailerCallbackSoap (Port):
http://localhost:8080/b...

OrderManagerPortType (Port Type):

- cancelOrder**
input: cancelOrder
parameters: parameters
output: cancelOrderResponse
- customerOrder**
input: customerOrder
parameters: parameters
output: customerOrderResponse
- SalesOrder**
parameters: SalesOrderFault
output: SalesOrderFault

RetailerCallbackPortType (Port Type):

- SendSalesOrderNotification**
input: salesOrderNotification
parameters: Document
output: salesOrderNotificationAck
parameters: Document

Outline (View):

- Imports
- Types
- Services
- Bindings
- Port Type
- Messages

Properties (View):

General
Name: OrderManagerPortTypeService
Prefix: tns
Target namespace: http://org.jboss.esb/quickstarts/bpel/ABI_OrderManager

Advanced...

BPEL Resources

<http://www.jboss.org/riftsaw>

Riftsaw Forums

WS-BPEL Primer:

<http://docs.oasis-open.org/wsbpel/2.0/Primer/wsbpel-v2.0-Primer.html>

WS-BPEL 2.0 Specification:

<http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html>

Apache Ode

<http://ode.apache.org/>



Overview

Downloads

Documentation

Community

Issue Tracker

Source Code

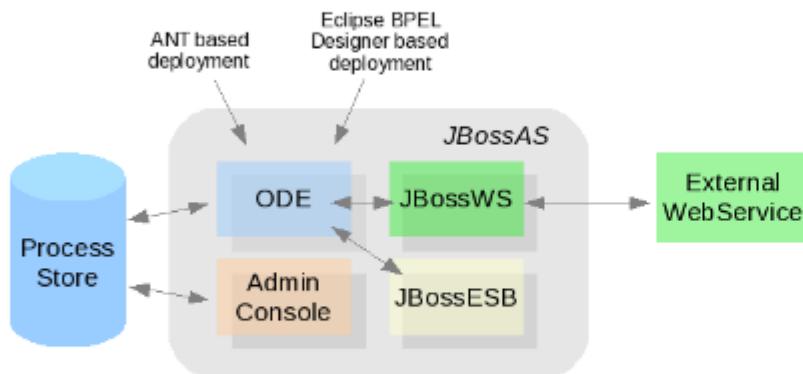
Build

Search project pages



JBoss BPEL Server

Project Riftsaw is a WS-BPEL 2.0 engine that is optimized for the JBoss Application Server container. WS-BPEL 2.0 is an XML-based language for defining business processes that orchestrate web services.



Riftsaw supports :

Useful Links

[Apache ODE](#)[WS-BPEL 2.0 Specification](#)[Eclipse BPEL Designer \(Bundled with JBoss Tools 3.1\)](#)[Screenshot Eclipse BPEL Designer](#)[Roadmap](#)[Slides of the July 22 Webinar](#)[Playback of the July 22 Webinar](#)[RiftSaw Blog](#)

Professional Enterprise Support

RESTful HTTP – SOA Platform 5 (Q1 2010)

CEP – SOA Platform 5 (Technology Preview)

Infinispan – SOA Platform 5.1 or 6.0 (TBD)

BPEL – BPEL Platform 2.0 (Q1 2010)

For more information at JBoss World 2009

ESB – Kevin Conner (Newcastle)

CEP via Drools Fusion – Edson Tirelli (Toronto)

Infinispan – Manik Surtani (London)

BPEL via Riftsaw – John Graham (Boston)

QUESTIONS?

TELL US WHAT YOU THINK:
REDHAT.COM/JBOSSWORLD-SURVEY

FOLLOW US:
TWITTER.COM/REDHATSUMMIT

TWEET ABOUT US:
ADD #SUMMIT AND/OR #JBOSSWORLD TO THE END
OF YOUR EVENT-RELATED TWEET

