

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

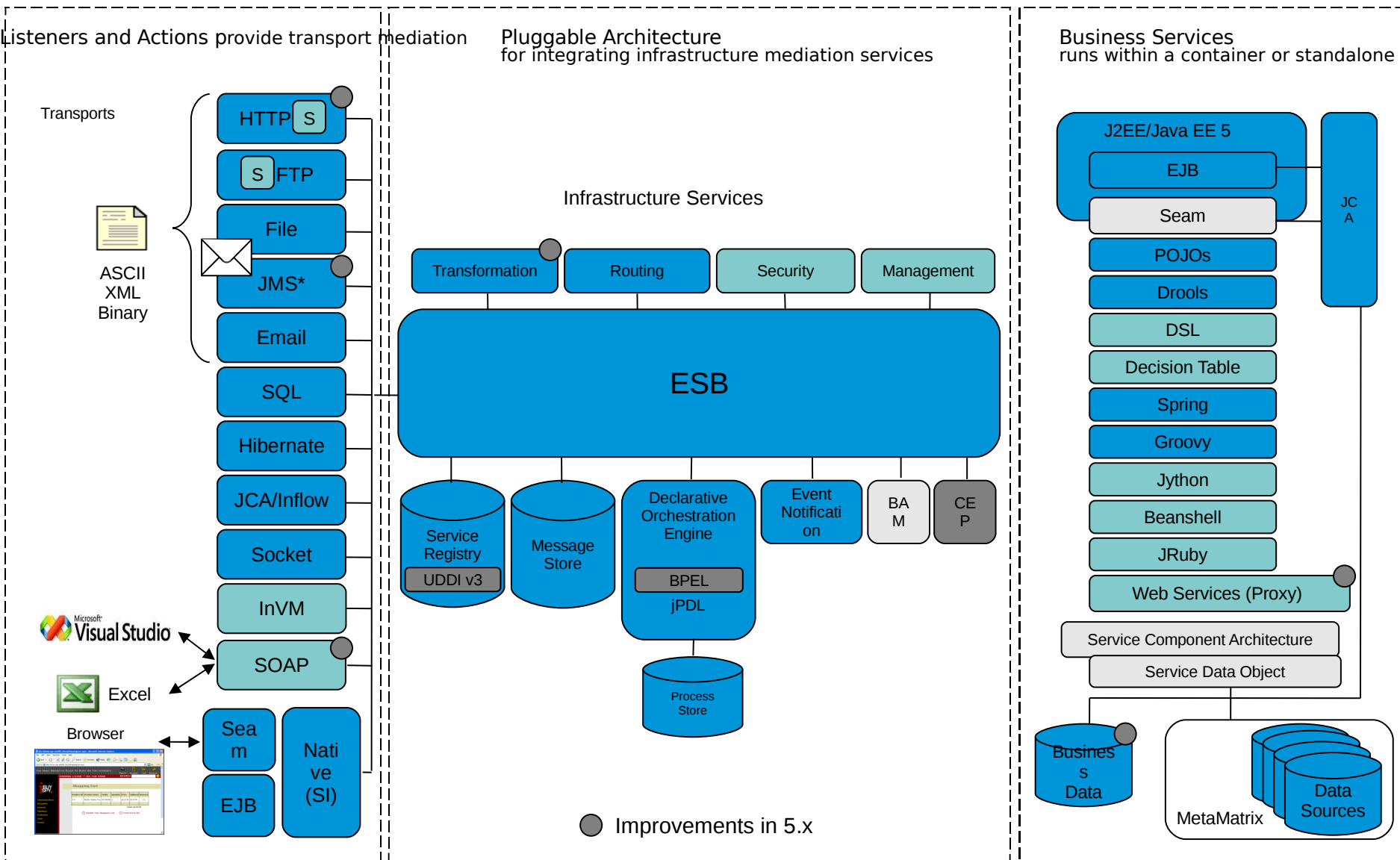




Event Listeners and Actions provide transport mediation

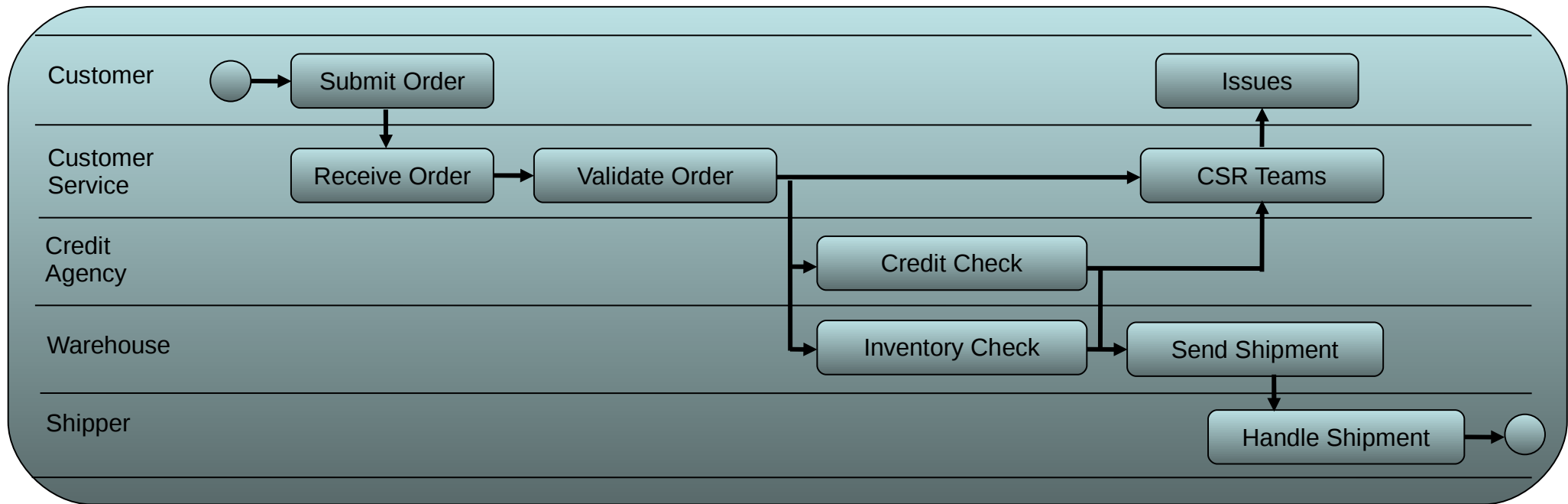
Pluggable Architecture
for integrating infrastructure mediation services

Business Services
runs within a container or standalone



JMS* - JBoss Messaging, IBM WebsphereMQ, TIBCO EMS, MRG-M

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

Inventory Check

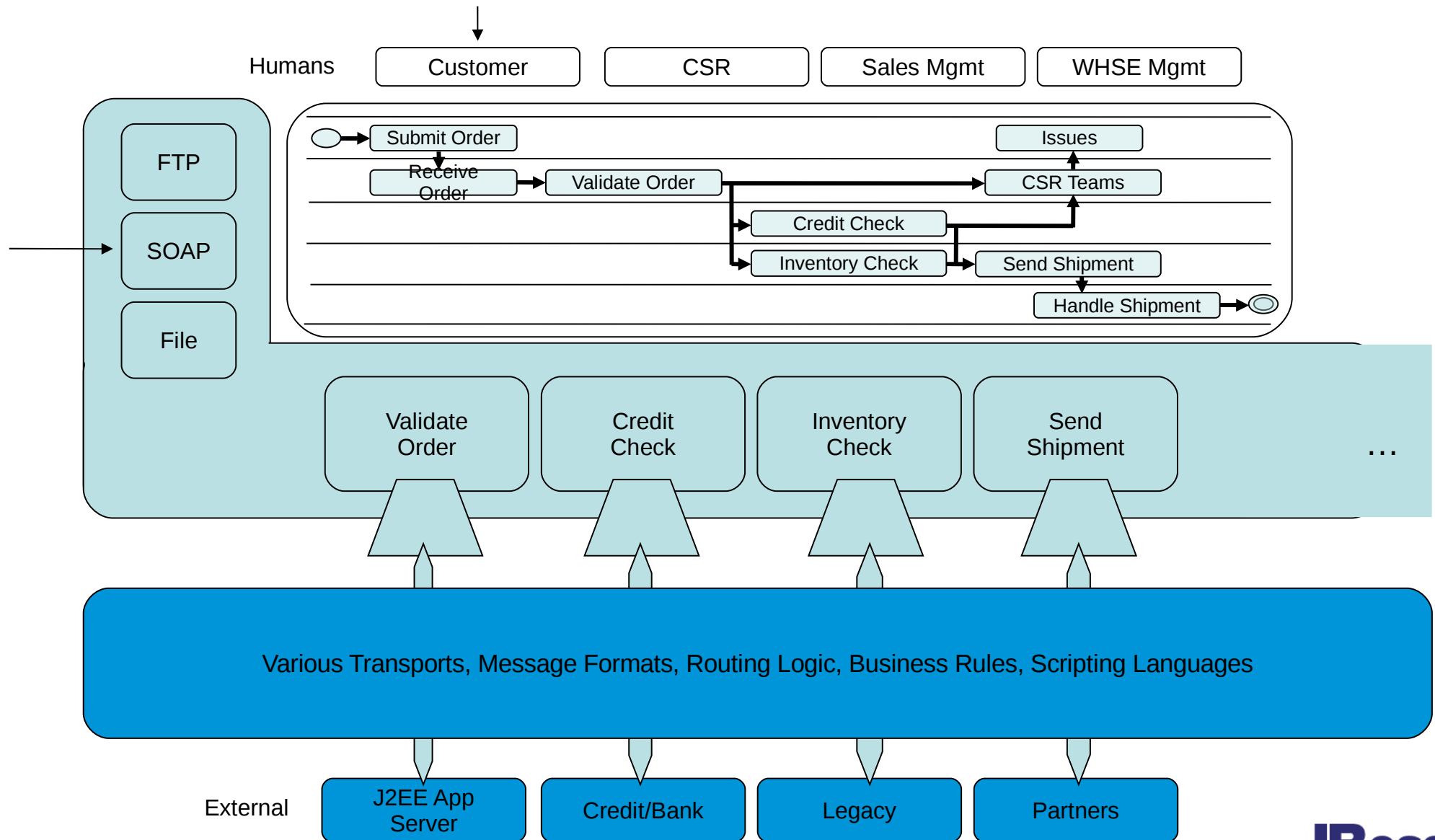
- a Send to N Warehouses
- b Handle N Responses
- c Determine Best WHSEs
- d Handle Drop-Ships

Send Shipment

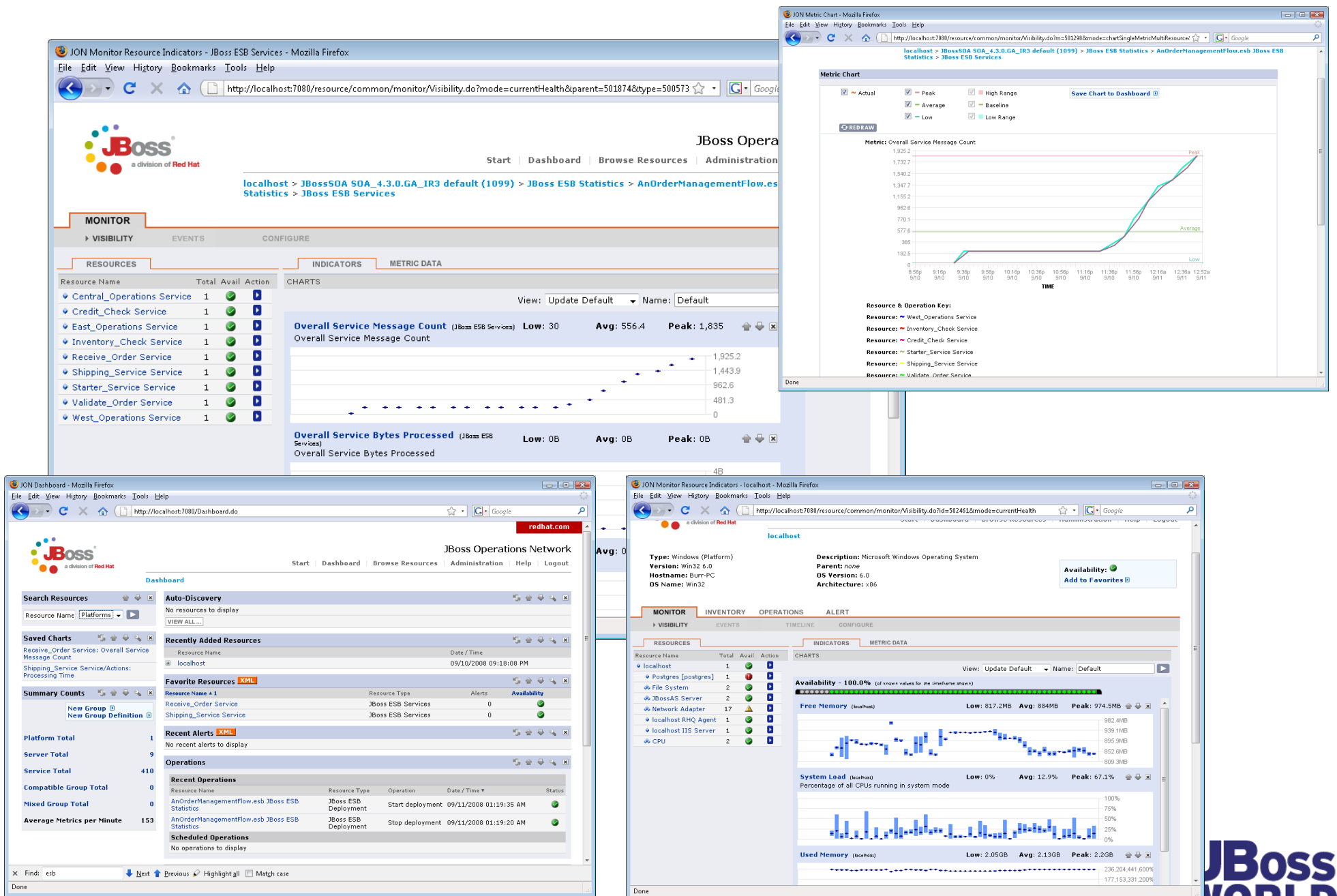
- a Determine Shipper(s)
- b Print Labels
- c Print Pick Tickets
- d Create & Send ASNs

**ESB Mediates
& Provides Services**

Synergy of ESB + BPM + Rules



SOA Monitoring and Management



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)

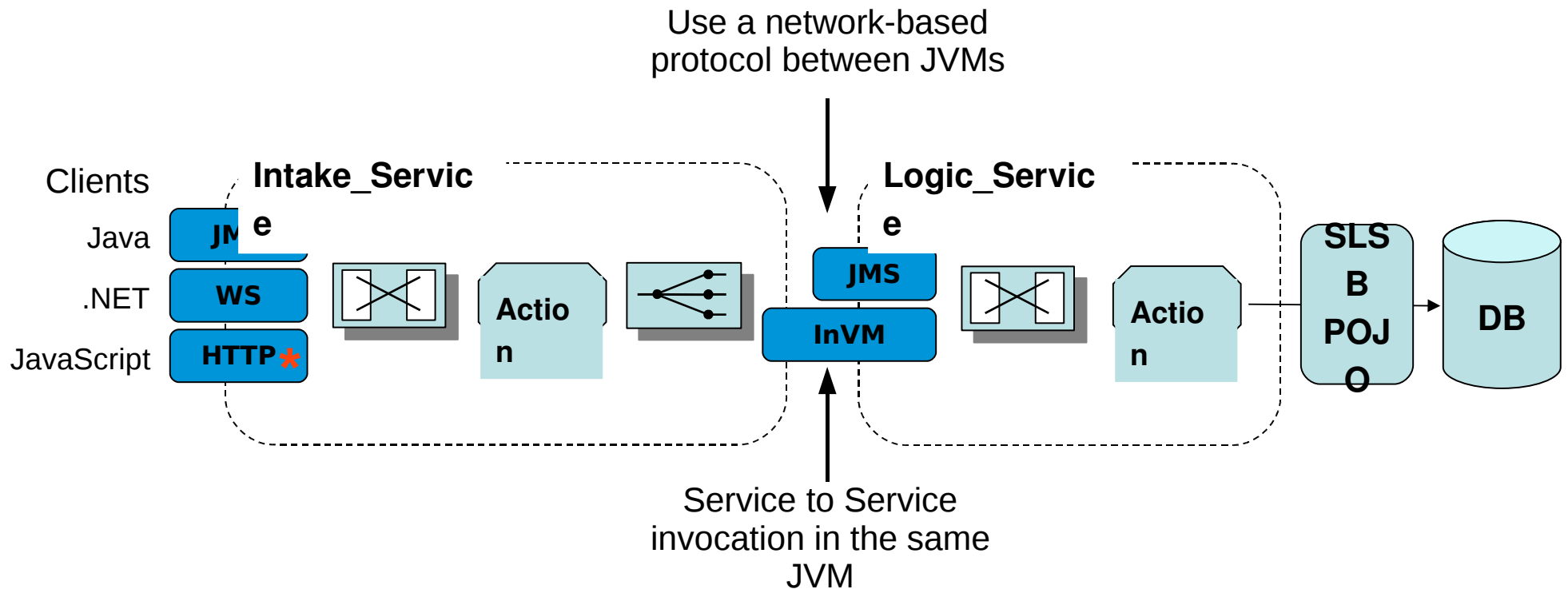
New HTTP Provider (jboss-esb.xml)

```
<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

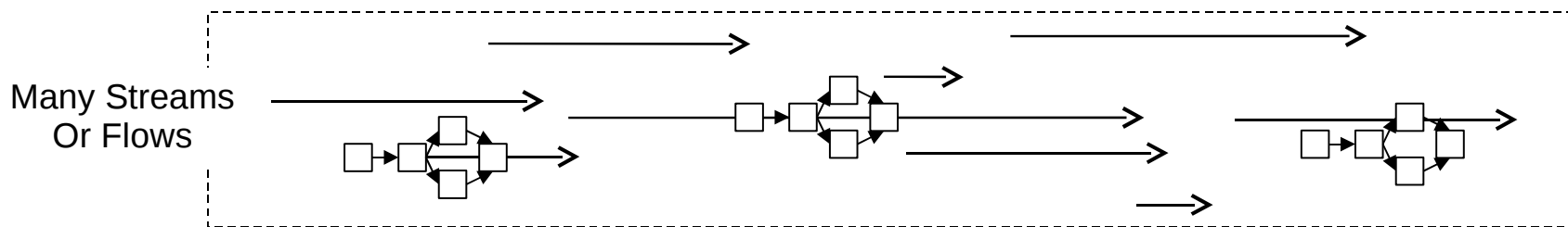


ATOM Demo

Complex Event Processing



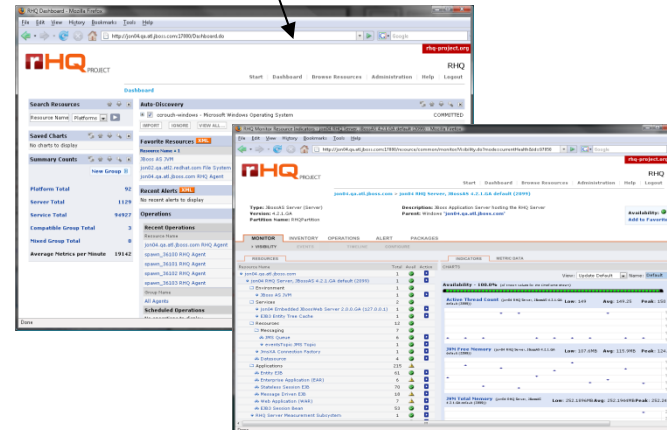
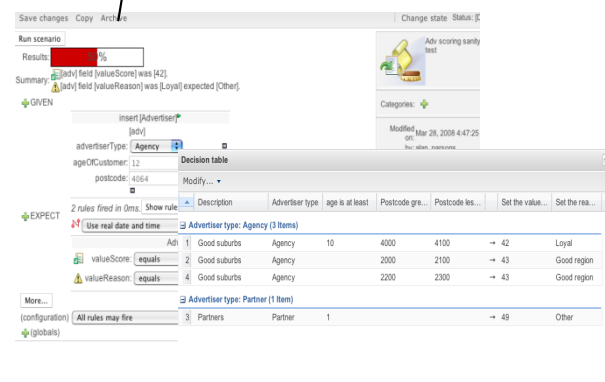
ESP + CEP via ESB, Rules & JON



ESB: consumption, capture, transformation, routing, orchestration
Rules: selection, aggregation, correlation, generation and publication



Governance Tools



Repository: for editing, versioning, testing and publishing new rules and SOA artifacts

JON: start/stop services, monitor and alert on service and action-level performance, monitor and alert on business metric values

Event Declaration & Selection

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

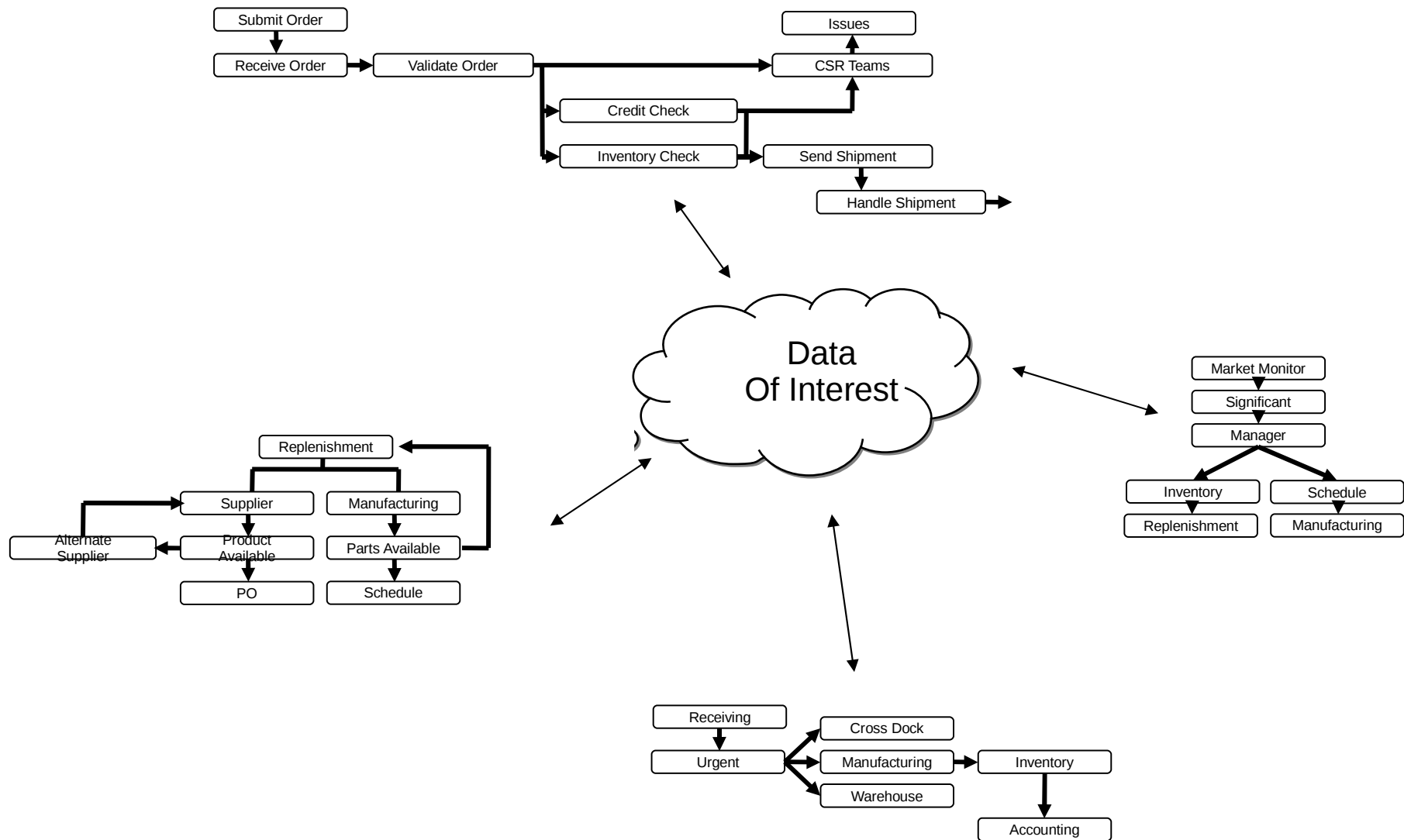
```
rule "average over last minute"
when
```

```
    $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

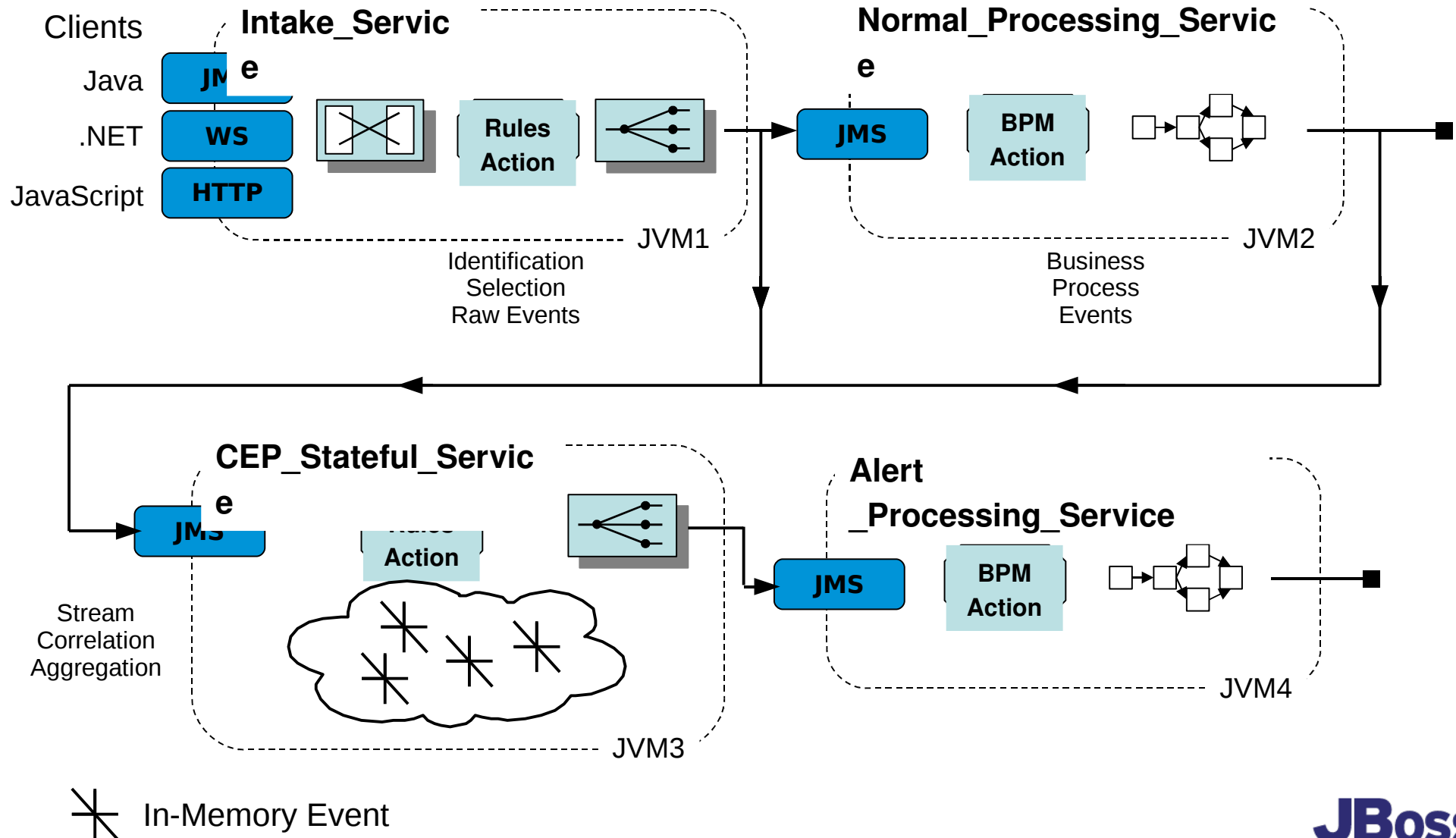
Stateful Rules Engines live in ESB Nodes, they retain the real-time history of the event flow

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

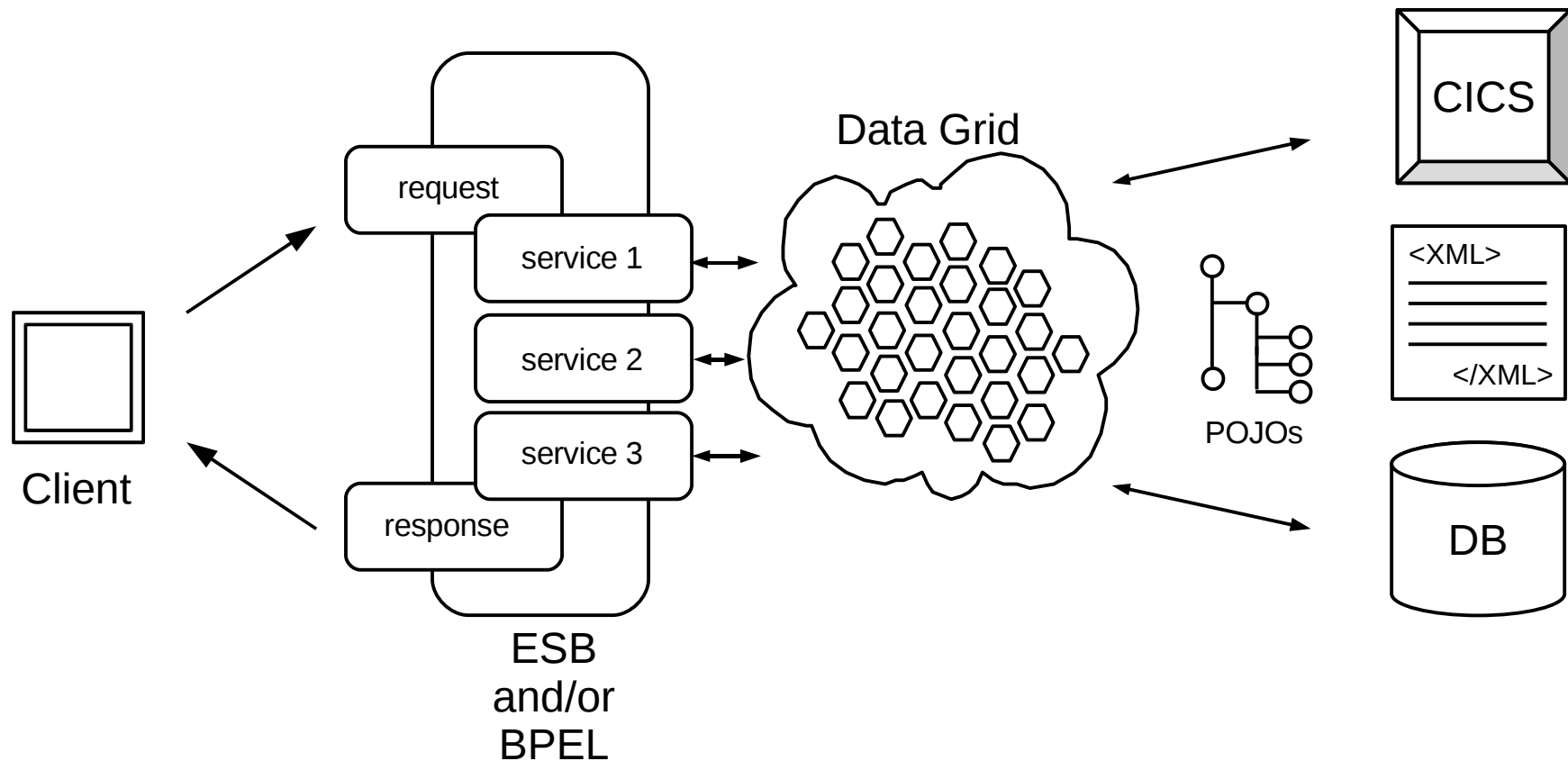
Improve fault tolerance, throughput, performance

Linear scalability

Predictable latency under increasing load

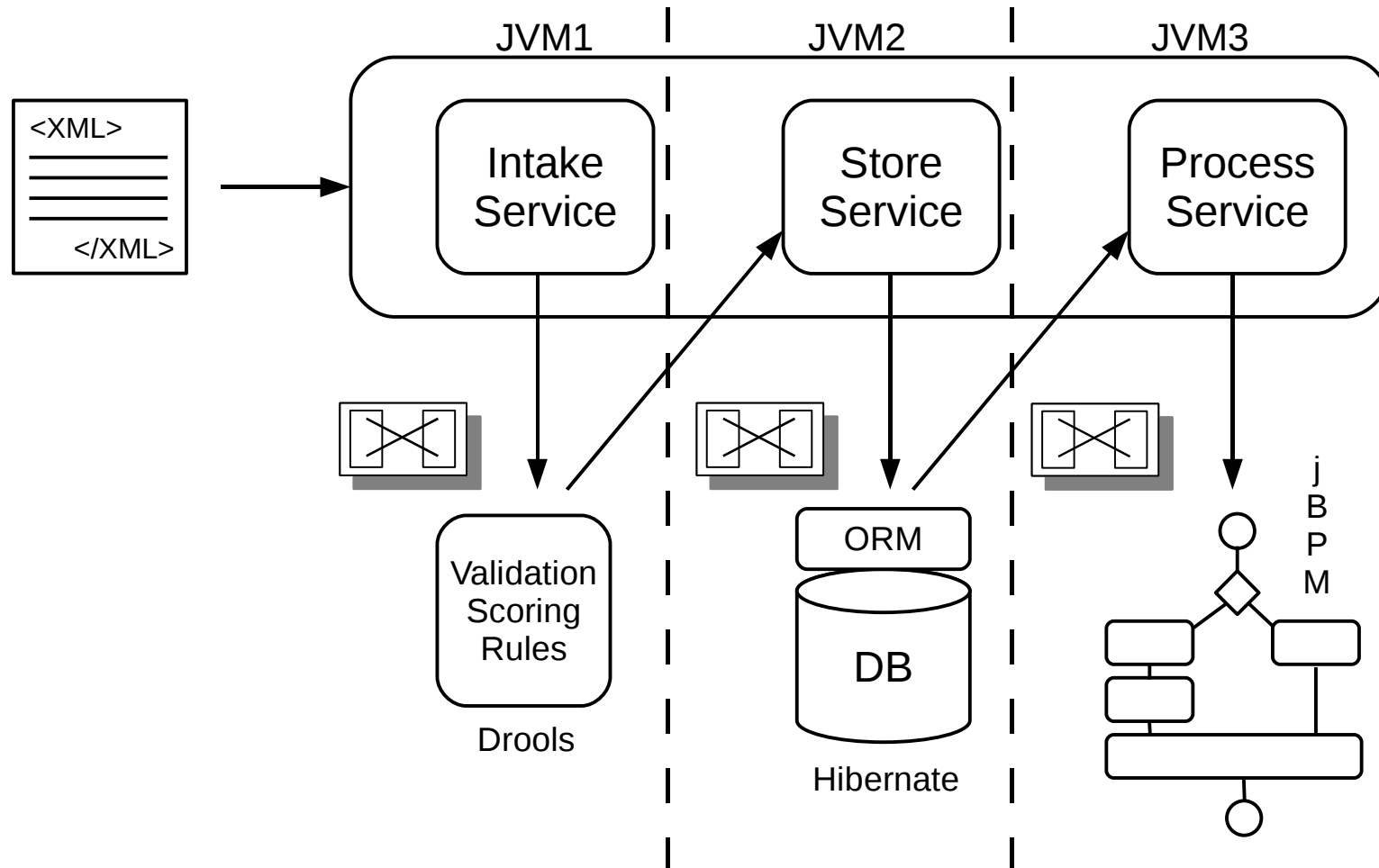
Access to more than a single JVM's Heap

Service Result Cache



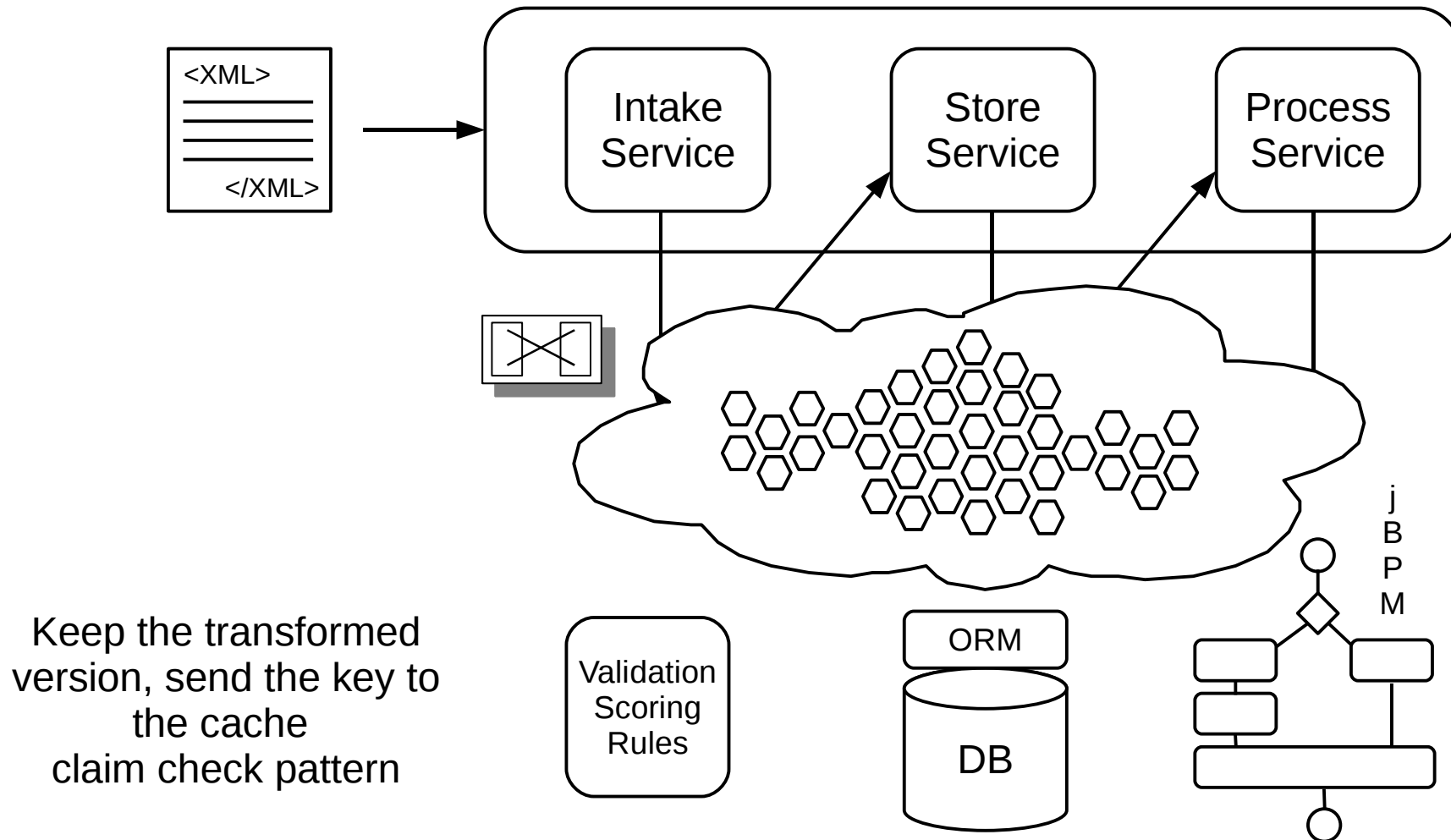
Results of service invocations to “expensive” resources
can be cached by the middle-tier architecture.
Transformations to POJOs can be time consuming

Hoppity, Hop, Hop



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

Still Hopping, less load

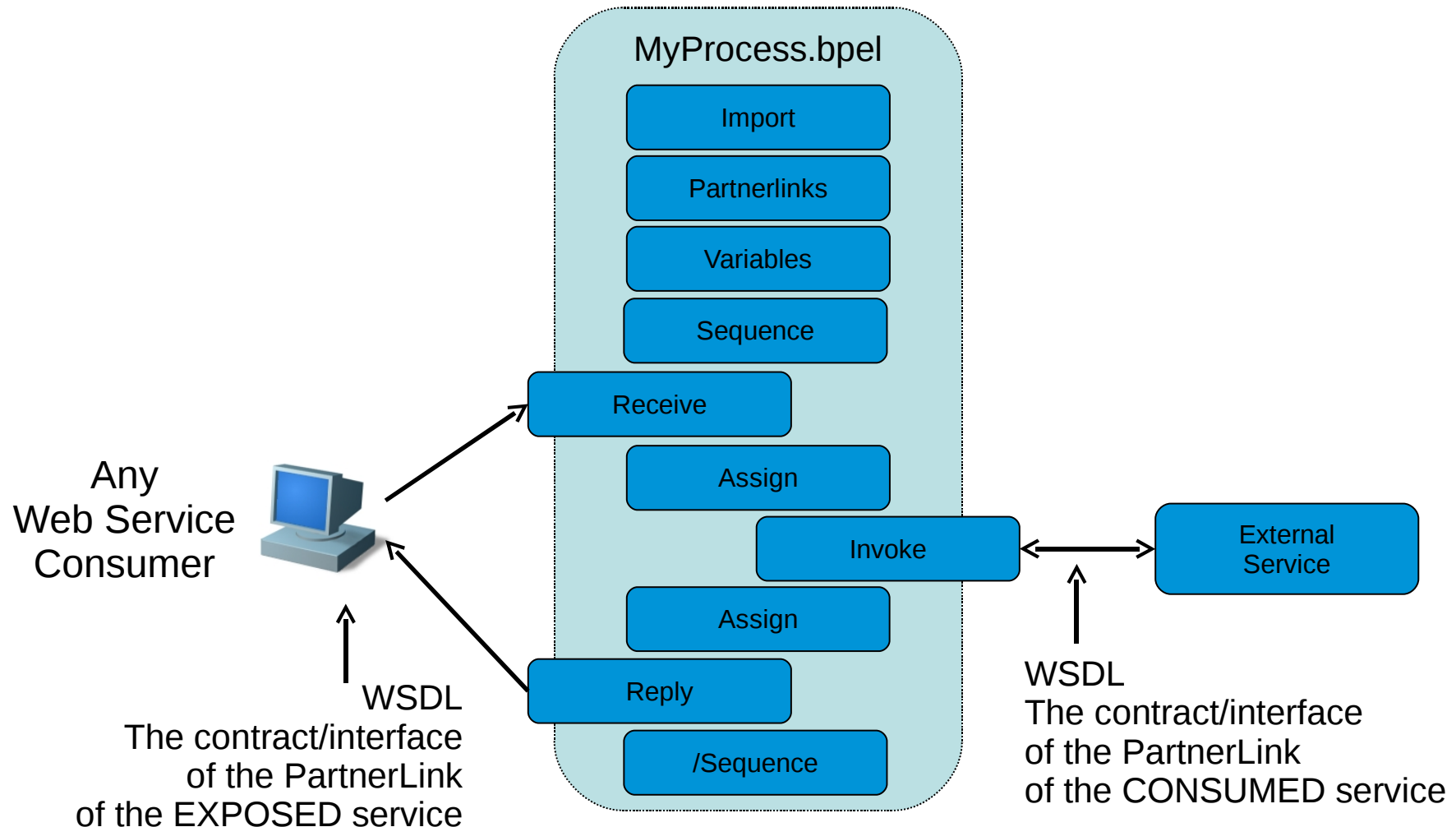


Data Grid Demo

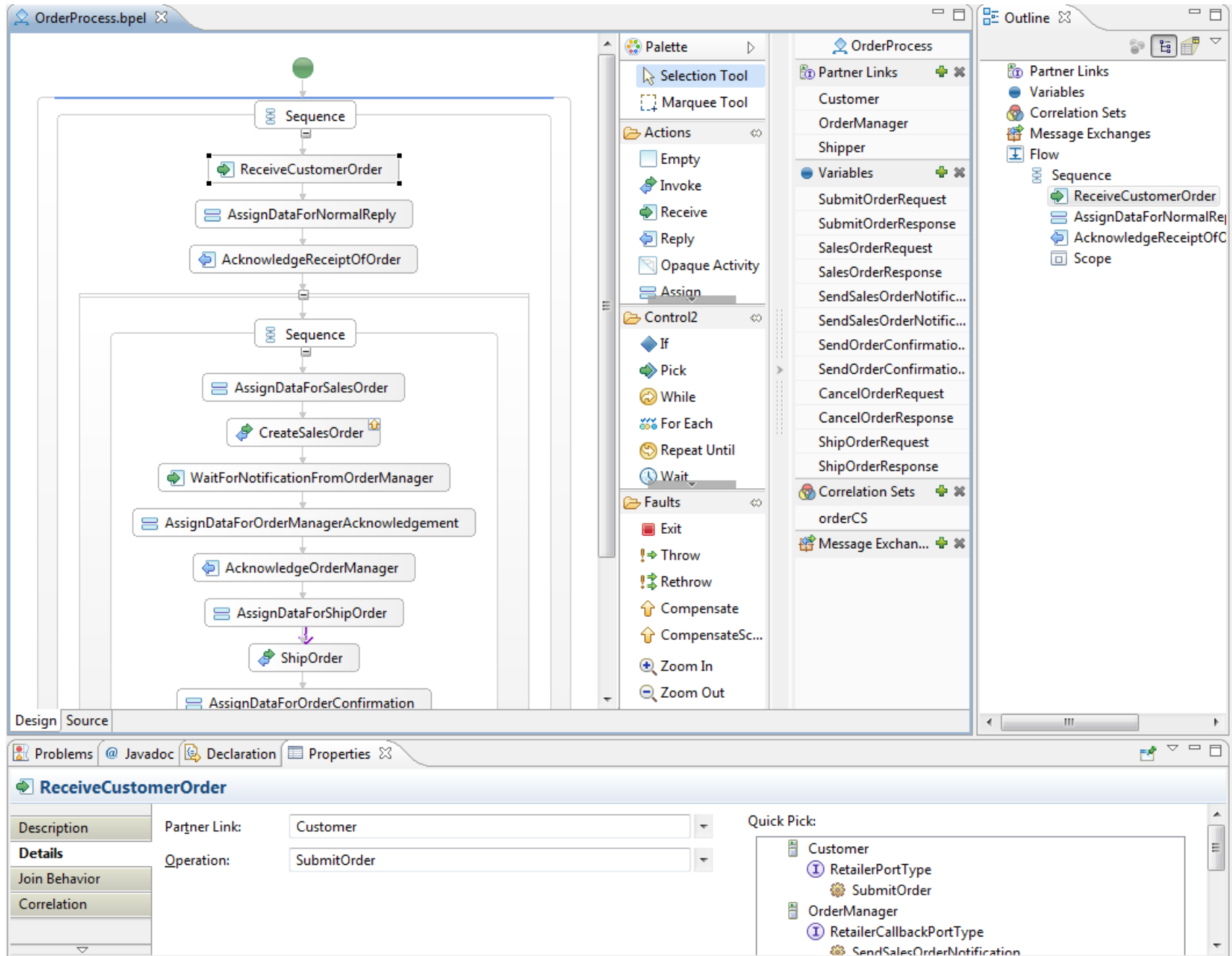
BPEL



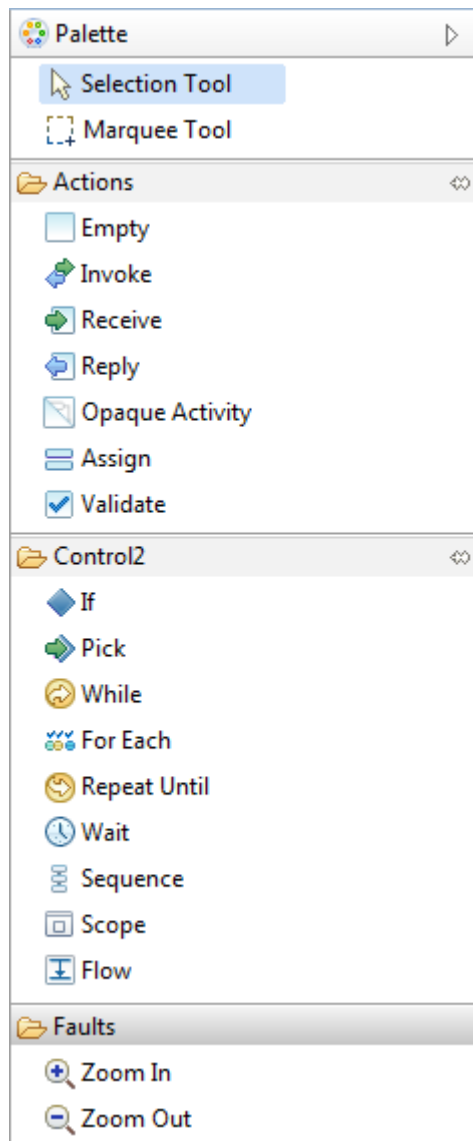
Simple BPEL Process Structure



Tools – BPEL Editor



BPEL Palette



Partner Link Properties

The screenshot displays the JBoss IDE interface for configuring a Partner Link. The left sidebar shows a project named 'OrderProcess.bpel' with a tree view containing 'Partner Links', 'Variables', 'Correlation Sets', and 'Message Exchanges'. The 'Partner Links' section is expanded, showing a list of roles: 'Customer', 'OrderManager', and 'Shipper'. The 'Customer' role is selected.

The main window is titled 'Customer' and shows the 'Properties' tab. The 'Description' section indicates the 'Partner Link Type' is 'PurchasingPLT'. The 'Details' section is divided into two main areas: 'My Role' and 'Partner Role'. In the 'My Role' section, the 'Buyer' role is selected. In the 'Partner Role' section, the 'Buyer' role is also selected, and the 'Initialize Role' checkbox is checked. The 'Documentation' section is currently empty.

The 'My Operations' section lists the following operations:

- SubmitOrder
 - SubmitOrderRequest
 - Document : customerOrder
 - header : OrderHeader
 - items : OrderItems
 - SubmitOrderResponse
 - Document : customerOrderAck

The 'Partner Operations' section lists the following operations:

- SendOrderConfirmation
 - SendOrderConfirmationRequest
 - Document : orderConfirmation
 - customerNumber : string
 - poNumber : string
 - orderNumber : string
 - SendOrderConfirmationResponse
 - Document : orderConfirmationAck

Tools – WSDL Editor

The screenshot displays the JBoss WSDL Editor interface. The top tab bar shows four open files: OrderProcess.bpel, BPELRetailer.wsdl, Customer.wsdl, and OrderManager.wsdl. The main workspace shows a diagram with two service boxes on the left: OrderManagerPortTypeService (with port OrderManagerPortTypePort at http://localhost:8865) and RetailerCallbackService (with port RetailerCallbackSoap at http://localhost:8080/b...). Arrows point from these services to their respective port type definitions in the center. The OrderManagerPortType definition includes two operations: cancelOrder and customerOrder. The RetailerCallbackPortType definition includes one operation: SendSalesOrderNotification. The right-hand Outline pane lists the project structure, including Imports, Types, Services, Bindings, Port Type, and Messages. The bottom pane shows the 'Messages' tab for the selected service, displaying fields for Name, Prefix, and Target namespace.

OrderManagerPortTypeService
OrderManagerPortTypePort
http://localhost:8865

RetailerCallbackService
RetailerCallbackSoap
http://localhost:8080/b...

OrderManagerPortType

Operation	Input	Output
cancelOrder	parameters	cancelOrderResponse
customerOrder	parameters	customerOrderResponse

RetailerCallbackPortType

Operation	Input	Output
SendSalesOrderNotification	Document	salesOrderNotificationAck

Outline

- Imports
- Types
 - http://org.jboss.esb/quicksta
- Services
 - RetailerCallbackService
 - OrderManagerPortTypeServi
- Bindings
 - RetailerCallbackSoap
 - OrderManagerPortTypeBind
- Port Type
 - OrderManagerPortType
 - RetailerCallbackPortType
- Messages
 - OrderManagerPortType_cust
 - SendSalesOrderNotificationF
 - OrderManagerPortType_can
 - SendSalesOrderNotificationF
 - OrderManagerPortType_cust
 - SalesOrderFault
 - OrderManagerPortType_can

Messages

General

Name: OrderManagerPortTypeService

Documentation

Prefix: tns

Extensions

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/>

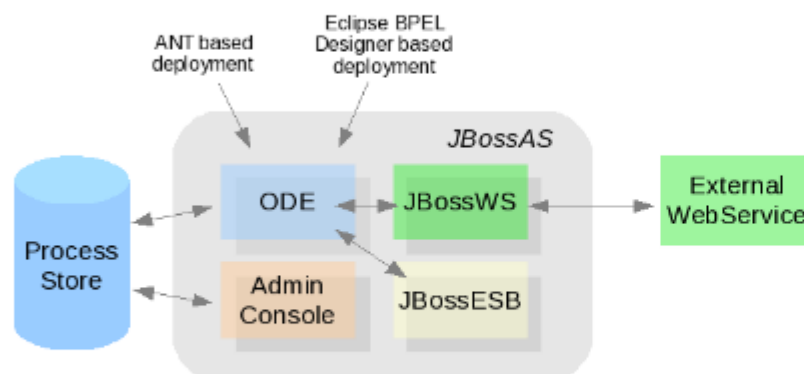


RS RiftSaw



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.



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](#)

Riftsaw supports :



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](https://redhat.com/jboss-world-survey)**

FOLLOW US:
[TWITTER.COM/REDHATSUMMIT](https://twitter.com/REDHATSUMMIT)

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