Paris JUG

10/05/2011

Sponsors
Platinium

www.parisjug.org

Xebia · zenika ARCHITECTURE INFORMATIQUE · OBJET DIRECT · NOVEDIA VISION · Sfeir Groupe · OXiane · Fast Connect · valtech · IBM ibm.com/developerWorks

Sponsors
Gold

in fine group · Objectif · So@t · ippon TECHNOLOGIES

JAVA USER GROUP

www.parisjug.org

10/05/2011

Intégration avec Camel et ServiceMix

Charles Moulliard

Architecte de Solutions

Apache Committer

FuseSource

A Progress Software Company

Objectif

« Faire découvrir les projets Apache orientés Intégration comme Camel ainsi que le conteneur ServiceMix et les architectures cibles»

Intervenant

- **Charles Moulliard**
  - Ingénieur Agronome & Licencié en Zoologie
  - 17 expérience dans le monde du développement IT (J2EE, .NET)
  - Spécialisé dans le monde des nouvelles technologies Web/Internet/Intégration
  - Chef de projet dans le monde bancaire, financier, telco, assurance, transports
  - Architecte de solutions chez FuseSource
  - Committeur Apache sur les projets : ServiceMix, Karaf (PMC) & Camel
  - BIO : « Apache ServiceMix and Karaf in Action » - Manning editor

Sommaire

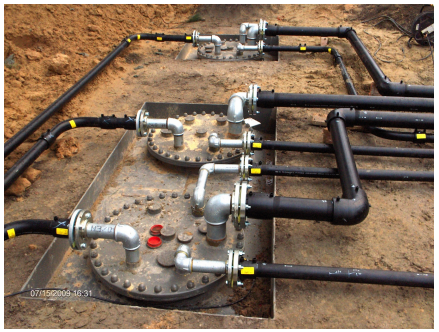
- **Intégration avec Camel et ServiceMix**
  - Démystifier l' ESB
  - ServiceMix - plateforme multi-conteneurs
  - Présentation des projets - Camel, CXF, Karaf et ActiveMq
  - Description des topologies possibles - messaging, osgi, messaging + osgi, web
  - Haute disponibilité, scalabilité et clustering - ActiveMQ, ServiceMix et ActiveMQ - ServiceMix

« Démystifier l' ESB et présenter la plateforme Apache ServiceMix »

ESB - ServiceMix

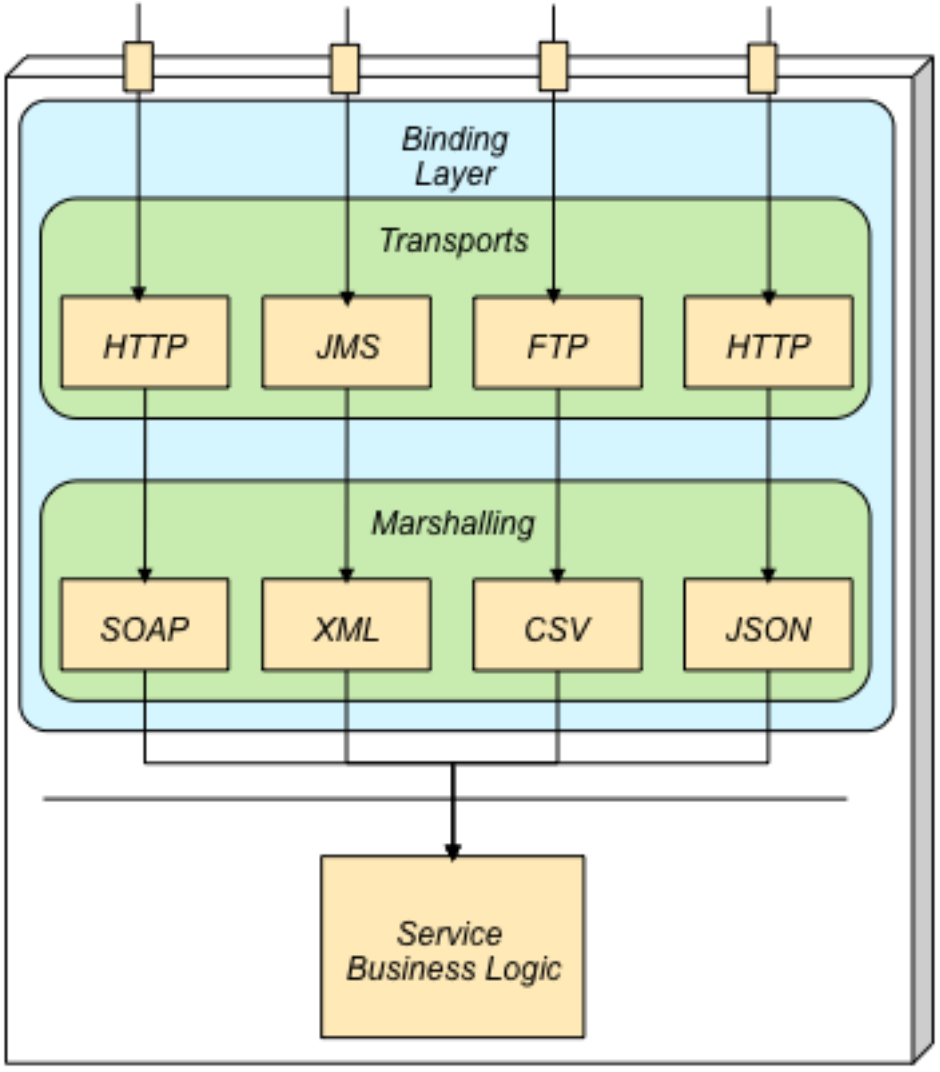
- Demystifier le concept - Enterprise Service Bus
- → Ce n'est pas un
- → Ni un

Mais un plateforme d' échange de messages, d'objets transitant via une couche de transport



ESB – ServiceMix (1)

- Permet de séparer la couche métier de celles qui vont **transporter** l'information et **transformer**



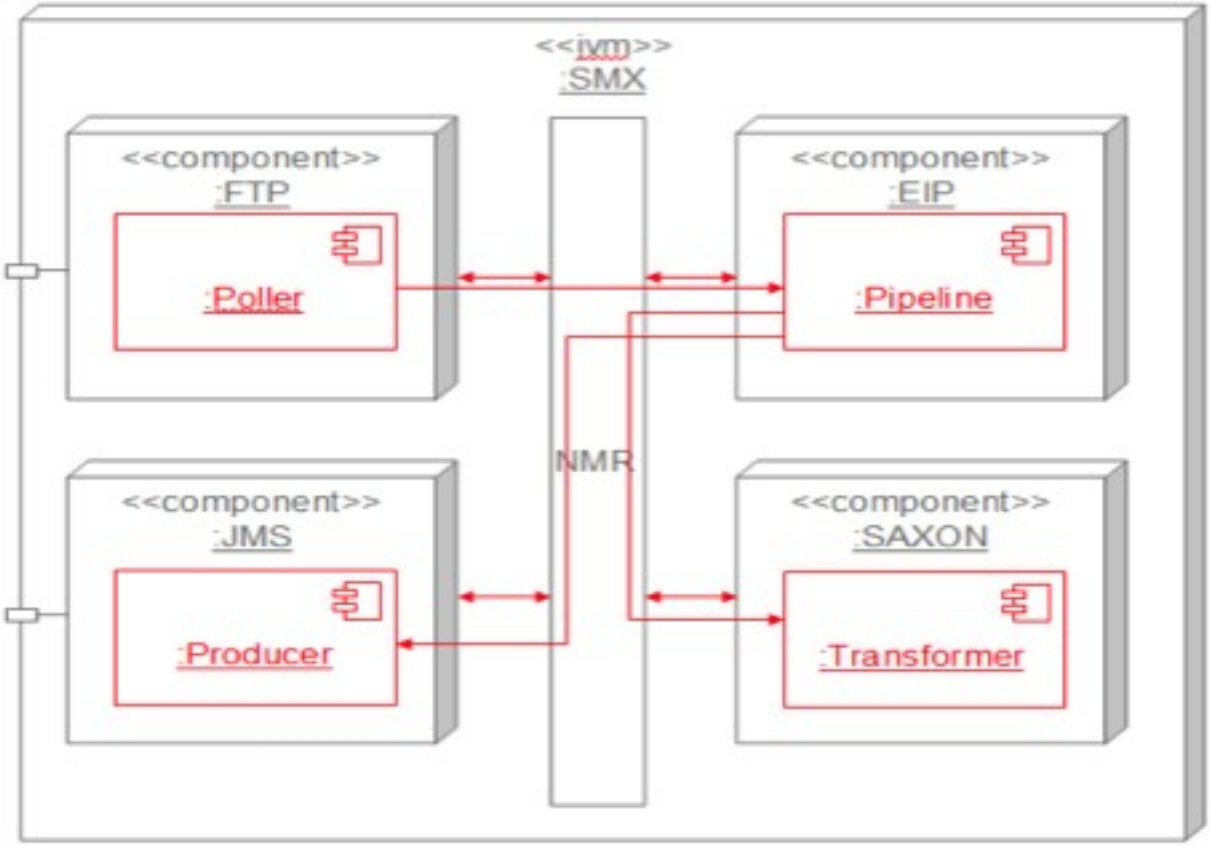
SMX – porte conteneurs

- ServiceMix – projet de la fondation Apache
- Créé en 2005 pour implémenter la spécification **J**ava **B**usiness **I**ntegration
- Découple les composants (métier, technique) via l'envoi de messages **XML**
- Défini un modèle de packaging/déploiement d' applications (**SU/SA**)
- Couche transport s'appelle - **N**ormalised **M**essaging **R**outer

SMX – porte conteneurs (2)

- Approche JBI est intéressante mais **contraignante**
  - Format XML
  - Routage encapsulé
  - Développement des composants
  - Peu de support de la part des acteurs (IBM, Oracle, …) pour faire évoluer la spécification
  - Parallèle avec EJB est intéressante
- Positionnement de ServiceMix pour le future

SMX – porte conteneurs (3)



Build an application by configuring and wiring endpoints as SUs, combining them into SAs that can be deployed atomically.

Endpoints are configured using xbean (Spring) configuration; deployment artifacts are created using maven plugins.

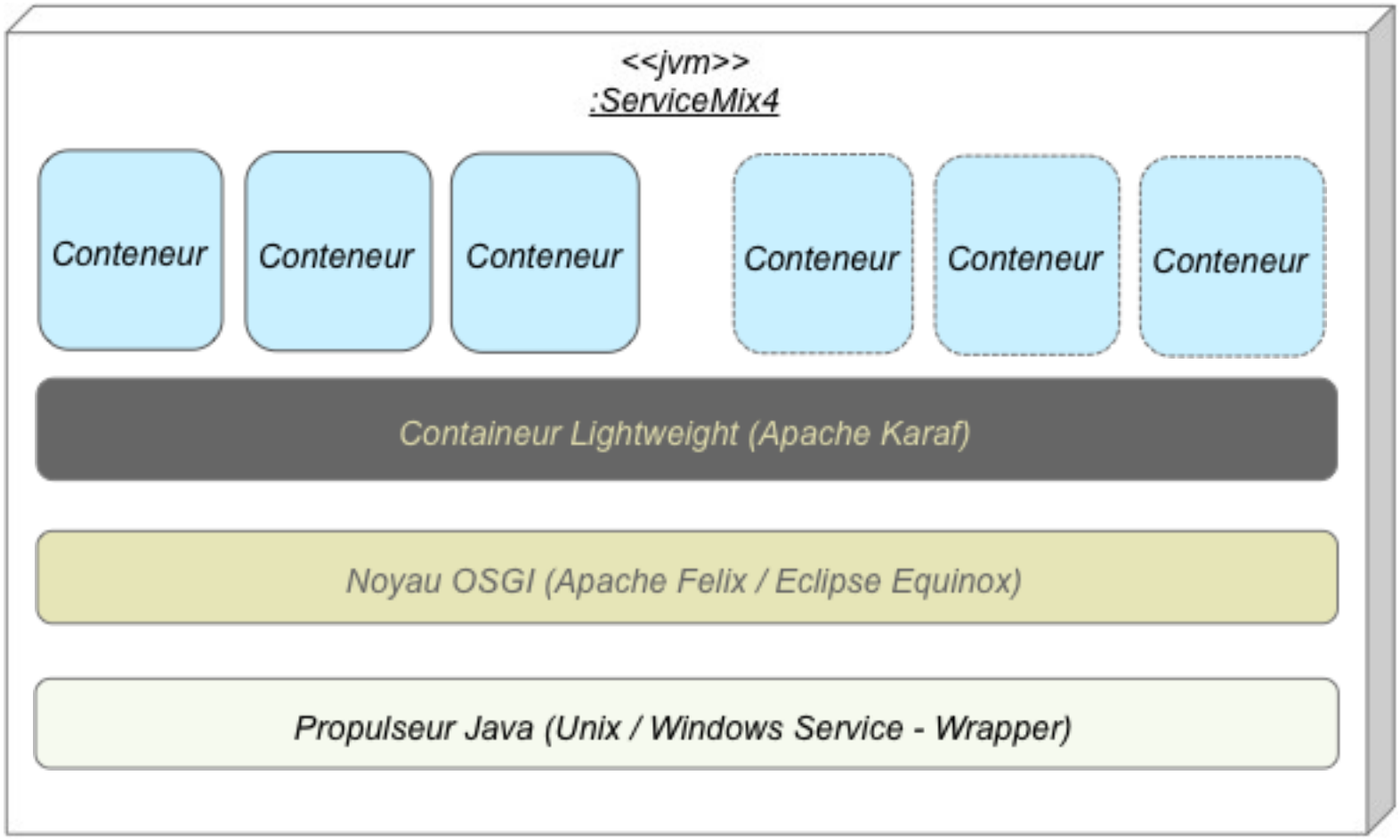
SMX – porte conteneurs (3)

- **ServiceMix 4**
- Basé sur un noyau **OSGI**
- Pq → offre modularité, gestion des jars, versioning
- Notre serveur devient un **conteneur** d' intégration opensource des projets
  - Camel
  - CXF
  - ActiveMQ, Aries (JPA, Transaction, JNDI), ….

Karaf – Cœur de ServiceMix

- **Date de naissance - 16th of June 2010**
- **Runtime** utilisant un serveur OSGI :
  - Apache Felix
  - Eclipse Equinox
- Fournit un **conteneur léger** dans lequel des
  - Applications
  - Composants
  - Routes, ...
- peuvent être déployés

Karaf – Cœur de ServiceMix



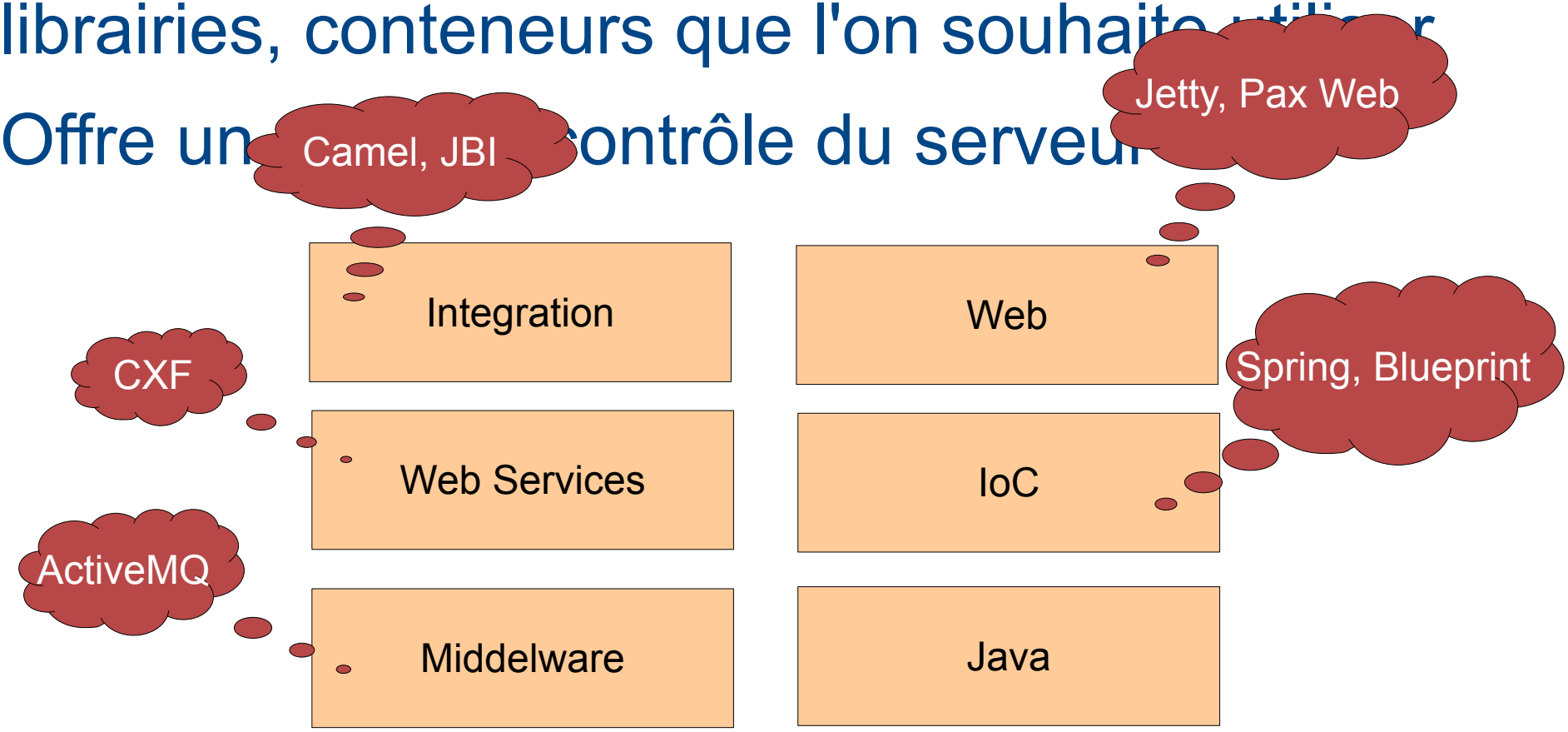
Karaf – Cœur de ServiceMix

Caractéristiques :

- Console d'administration (locale, remote, ssh, web, jmx)
- Systéme de provisionnement (features)
- Déploiement à chaud et gestion dynamiques des configs
- Gestion des instances
- Sécurité intégrée (JAAS → ldap, jdbc, file)
- Gestion centralisée des logs (log4j, logger, commons logging, ….)

Conteneur(s)

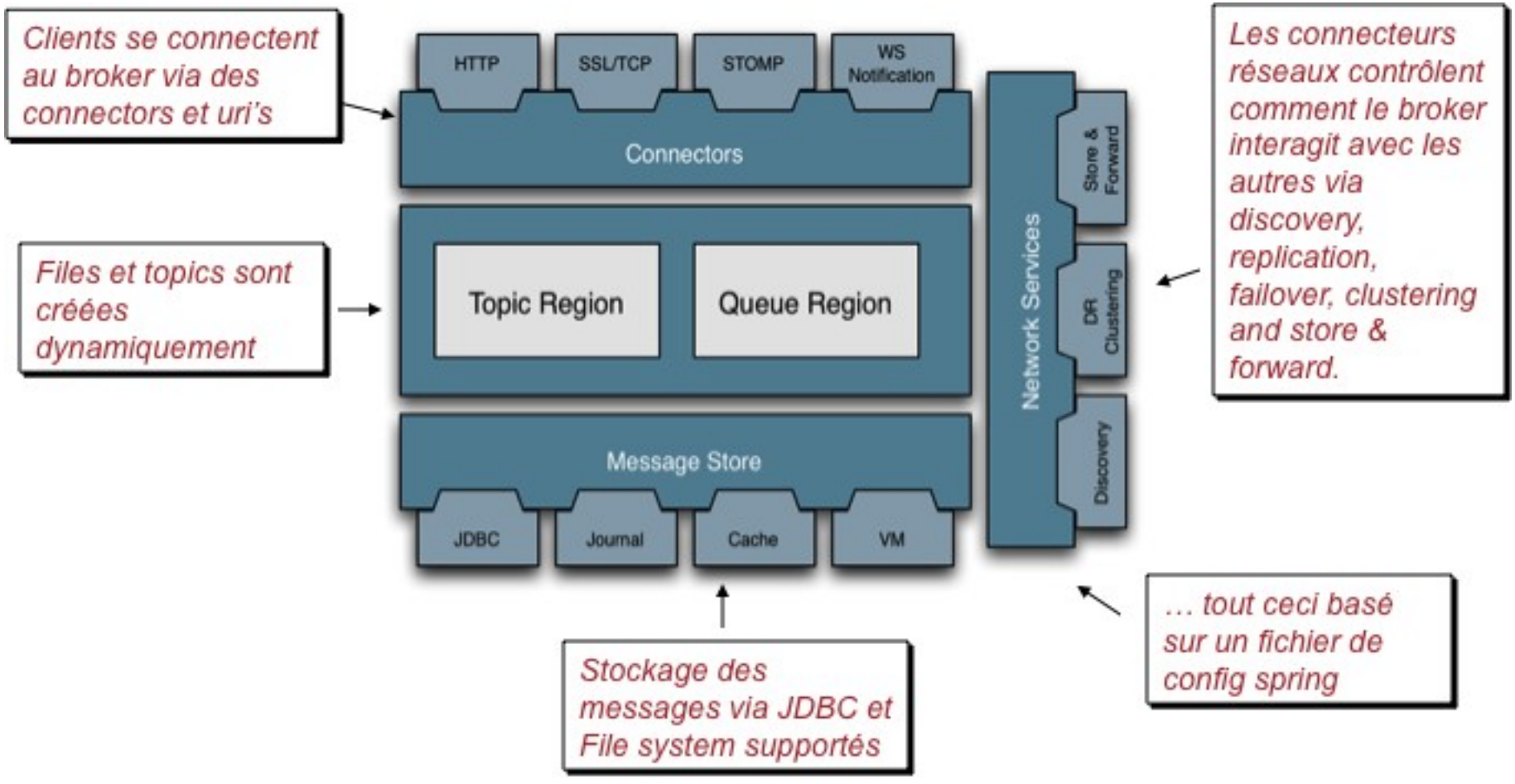
- Plateforme est modulable car on sélectionne les librairies, conteneurs que l'on souhaite utiliser
- Offre un contrôle du serveur



« ActiveMQ – l'agent d'échange / de persistence des messages »

ActiveMQ

- Une usine de fabrication de messages (PtP, Pub/Sub) supportant JMS et proposant connecteurs C, C++, .Net, PhP.



« CXF – L'outil de fabrication des Web Services »

CXF

- Fusion de 2 projets (Celtic and Xfire)
- Simplifie création et déploiement des web services via approche code-first ou WSDL-first.
- Supporte les standards comme :
  - JAX-WS : Web Services (XML/SOAP)
  - JAX-RS : RESTfull service (JSON)
  - SOAP 1.1, 1.2, WSDL 1.1
  - WS-Security : sécurise la connexion client serveur et vice versa
  - WS-Addressing : standardise les données échangées dans les entêtes SOAP pour le routage
  - WS-RM : garantie de livraison du message

CXF (1)

- Se configure trés facilement via plugin maven

```
            <plugin>
                <groupId>org.apache.cxf</groupId>
                <artifactId>cxf-codegen-plugin</artifactId>
                <version>${cxf-version}</version>
                <executions>
                    <execution>
                        <id>generate-sources</id>
                        <phase>generate-sources</phase>
                        <configuration>
                            <sourceRoot>$
{basedir}/target/generated/src/main/java</sourceRoot>
                            <wsdlOptions>
                                <wsdlOption>
                                    <wsdl>$
{basedir}/src/main/resources/report_incident.wsdl</wsdl>
                                </wsdlOption>
                            </wsdlOptions>
                        </configuration>
                        <goals>
                            <goal>wsdl2java</goal>
                        </goals>
                    </execution>
                </executions>
            </plugin>
```

CXF (2)

- Fichier Spring XML

```
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:cxf="http://camel.apache.org/schema/cxf"
       xsi:schemaLocation="
       http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd
       http://camel.apache.org/schema/cxf
http://camel.apache.org/schema/cxf/camel-cxf.xsd">

    <import resource="classpath:META-INF/cxf/cxf.xml"/>
    <import resource="classpath:META-INF/cxf/cxf-extension-soap.xml"/>
    <import resource="classpath:META-INF/cxf/cxf-extension-http-jetty.xml"/>

    ….

   <cxf:cxfEndpoint
        id="reportIncident"
        address="http://localhost:9080/incident"
        wsdlURL="etc/report_incident.wsdl"
        serviceClass="org.apache.camel.example.ReportIncidentEndpoint">
    </cxf:cxfEndpoint>

   …
</beans>
```

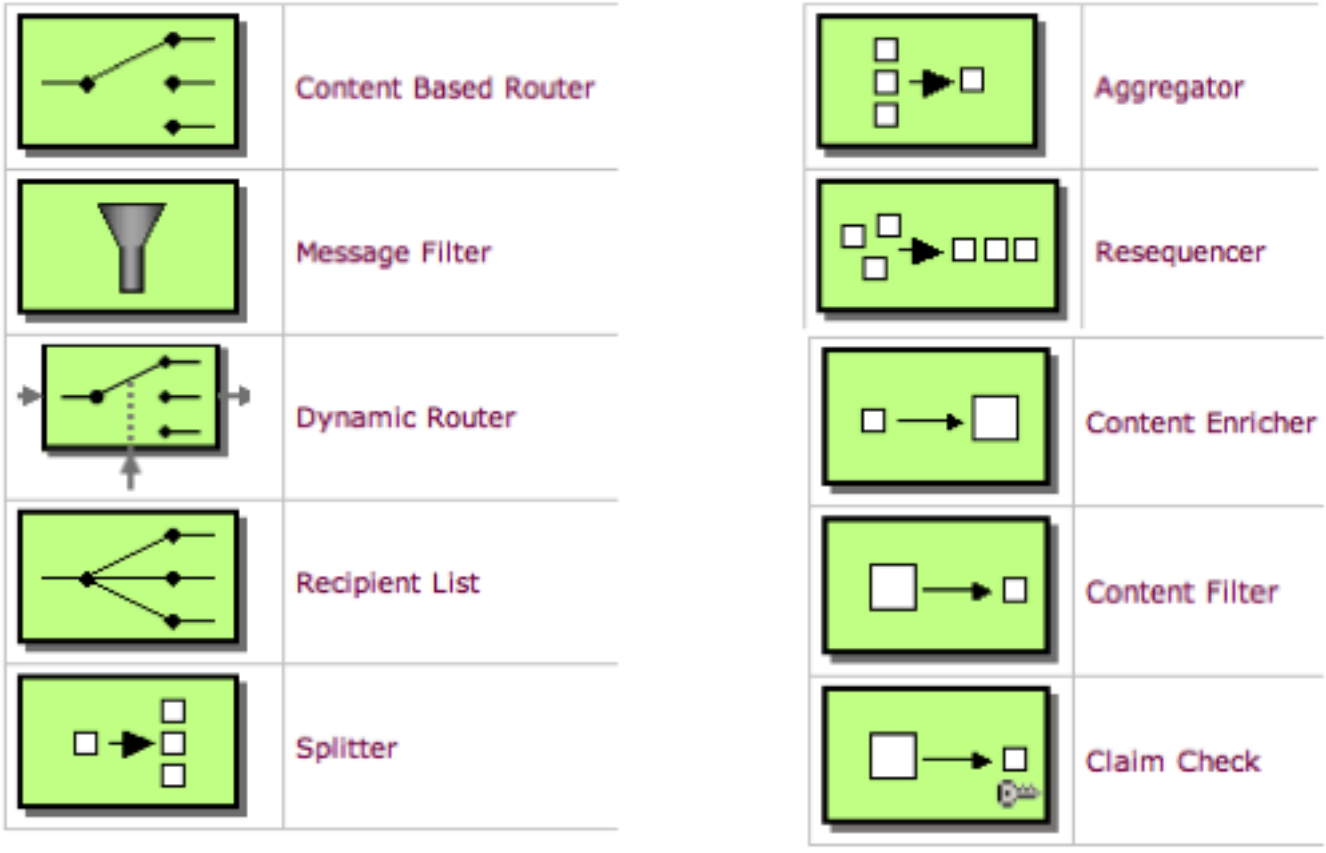
« Camel – L'intégrateur opensource des modéles entreprises (EIP) »

Camel

- Date de naissance Mars 2007
- Framework d'intégration opensource implémentant les modelés EIP
- Utilise un language specifique appelé DSL
- Supporte:
  - Architecture transactionnelle,
  - Echanges Sync/async,
  - Gestion des threads,
  - Mecanisme de controles des erreurs

Camel

- Plus de 50 modèles d'intégration



- http://camel.apache.org/enterprise-integration-patterns.html

Camel

- Plus de 100 composants

activemq	crypto	flatpack	irc	ldap
activemq-journal	cxf	freemarker	javaspace	mail/imap/pop3
amqp	cxfrs	ftp/ftps/sftp	jbi	mina
atom	dataset	gae	jcr	mock
bean	direct	hdfs	jdbc	msv
bean validation	esper	hibernate	jetty	nagios
browse	event	hl7	jms	netty
cache	exec	http	jpa	nmr
cometd	file	ibatis	jt/400	printer

http://camel.apache.org/components.html

Camel

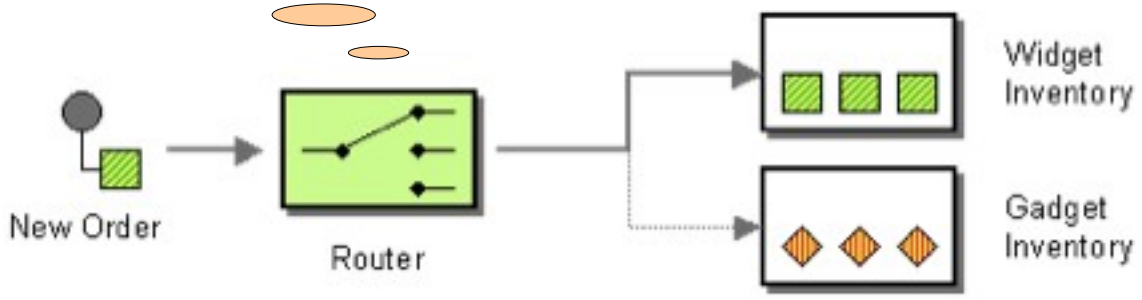
- 18 formateurs de données

bindy	protobuf
castor	serialization
csv	soap
crypto	tidy markup
flatpack	xml beans
gzip	xml security
hl7	xstream
jaxb	zip
json	dozer

http://camel.apache.org/data-format.html

Camel

Le modèle – Content Based Router



Langage DSL

```
from("activemq:topic:Quotes")
  .filter().xpath("/quote/product = 'widget'").
    to("bean:WidgetQuotes")
  .filter().xpath("/quote/product = 'gadget'").
    to("bean:GadgetQuotes");
```

Camel

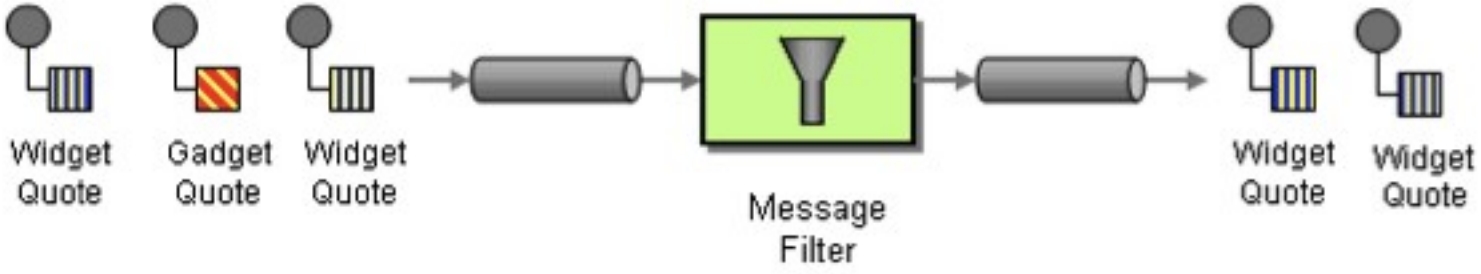
ActiveMQ

WebSphereMQ

from A

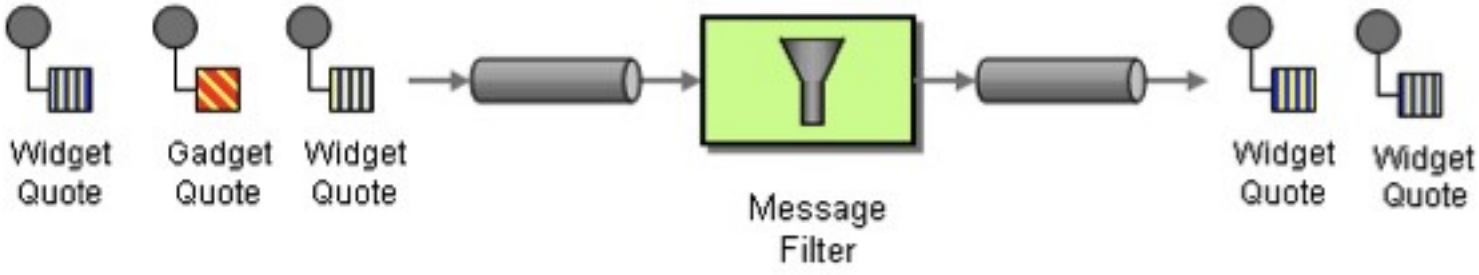
filter message

send to B



Camel

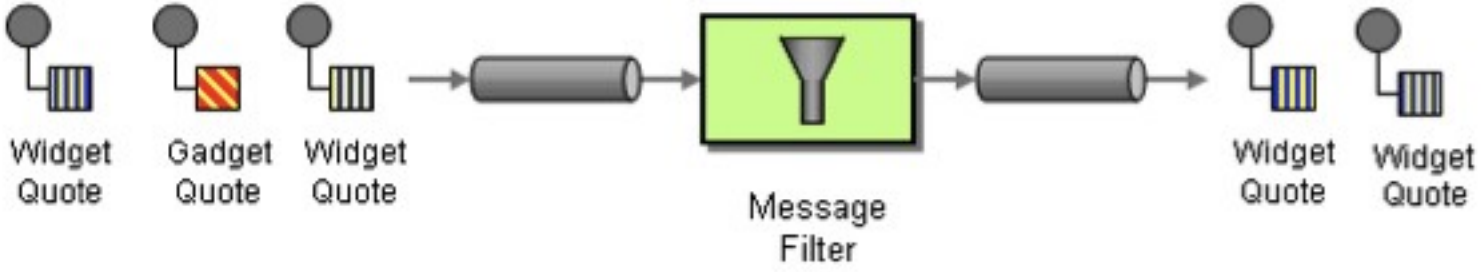
```
from(A).filter(isWidget).to(B);
```



Camel

```
isWidget = xpath("/quote/product = 'widget'");

from(A).filter(isWidget).to(B);
```



Camel


```
Endpoint A = endpoint("activemq:queue:quote");

Endpoint B = endpoint("mq:quote");

Predicate isWidget = xpath("/quote/product = 'widget'");


from(A).filter(isWidget).to(B);
```

Camel

- Java DSL


```
public void configure() throws Exception {
    Endpoint A = endpoint("activemq:queue:quote");
    Endpoint B = endpoint("mq:quote");
    Predicate isWidget = xpath("/quote/product = 'widget'");

    from(A).filter(isWidget).to(B);
}
```

Camel

- Route Camel + Java DSL


```
import org.apache.camel.builder.RouteBuilder;

public class FilterRoute extends RouteBuilder {

  public void configure() throws Exception {
    Endpoint A = endpoint("activemq:queue:quote");
    Endpoint B = endpoint("mq:quote");
    Predicate isWidget = xpath("/quote/product = 'widget'");

    from(A).filter(isWidget).to(B);
  }
}
```

Camel

- Une route Camel – EIP Filter Pattern

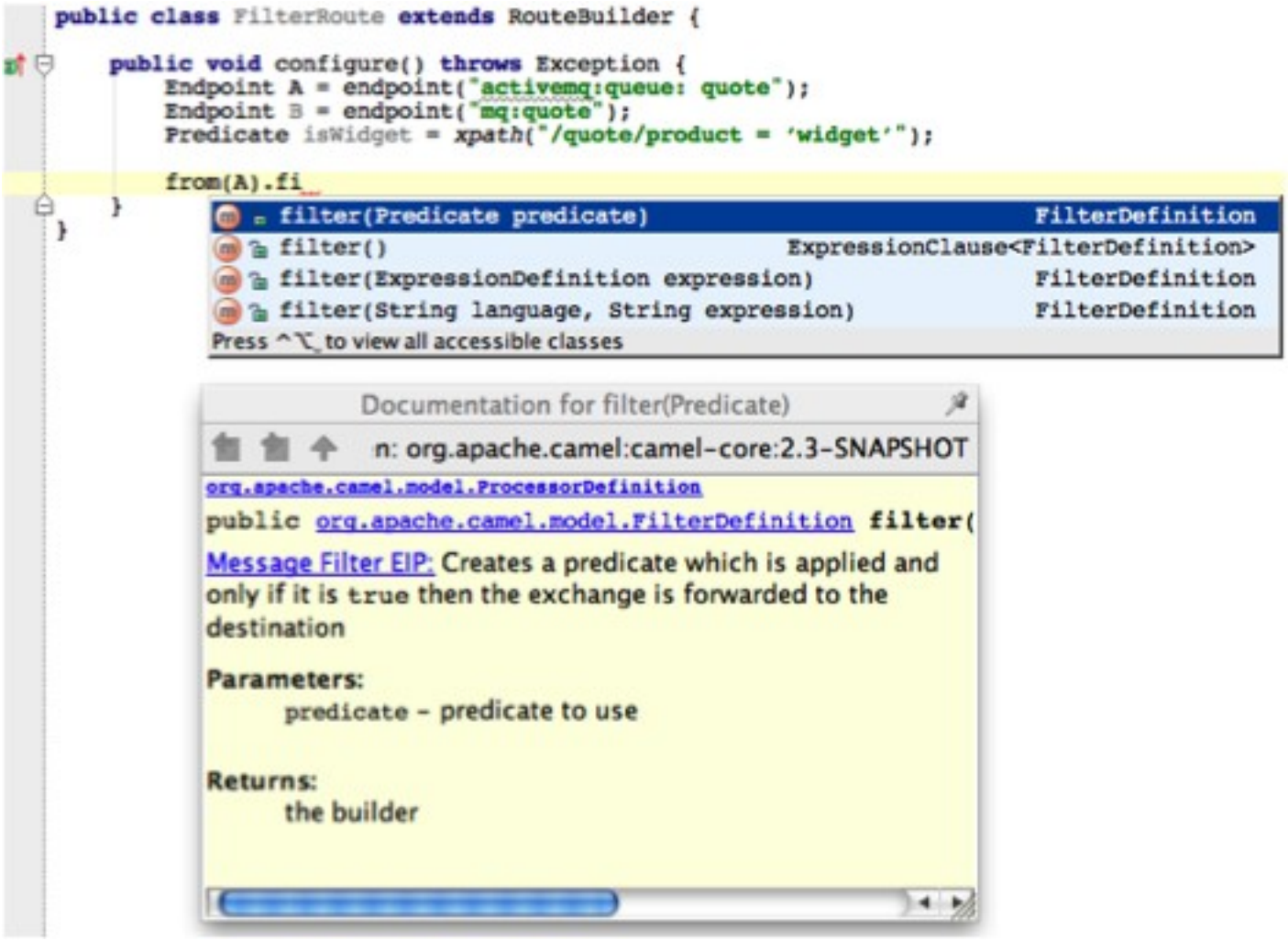

```
import org.apache.camel.builder.RouteBuilder;

public class FilterRoute extends RouteBuilder {

  public void configure() throws Exception {
    from("activemq:queue:quote")
      .filter().xpath("/quote/product ='widget'")
        .to("mq:quote");
  }
}
```

Camel

- Integration avec Eclipse, IntelliJ



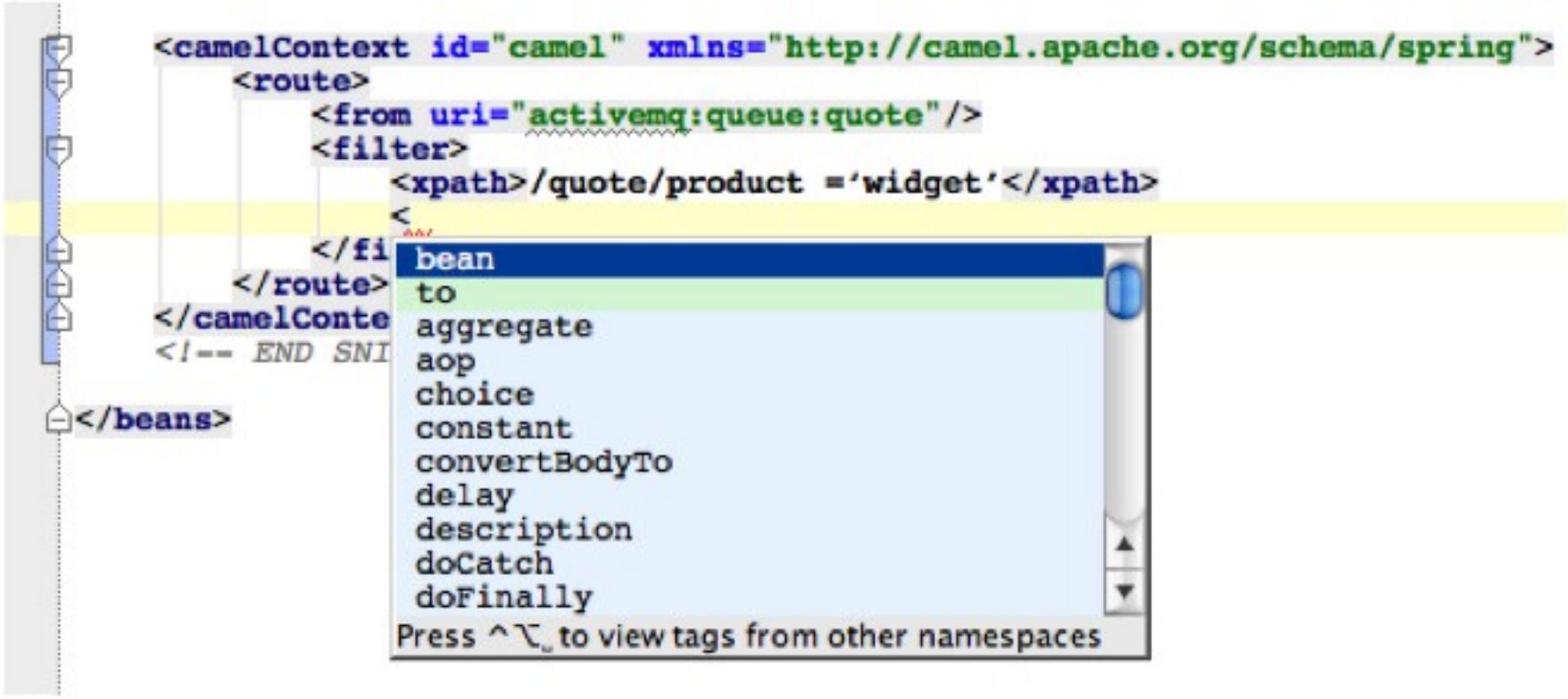
Camel

- Spring DSL

```
<camelContext>
  <route>
    <from uri="activemq:NewOrders"/>
    <choice>
      <when>
        <xpath>/order/product = 'widget'</xpath>
        <to uri="activemq:Orders.Widgets"/>
      </when>
      <otherwise>
        <to uri="activemq:Orders.Gadgets"/>
      </otherwise>
    </choice>
  </route>
</camelContext>
```

Camel

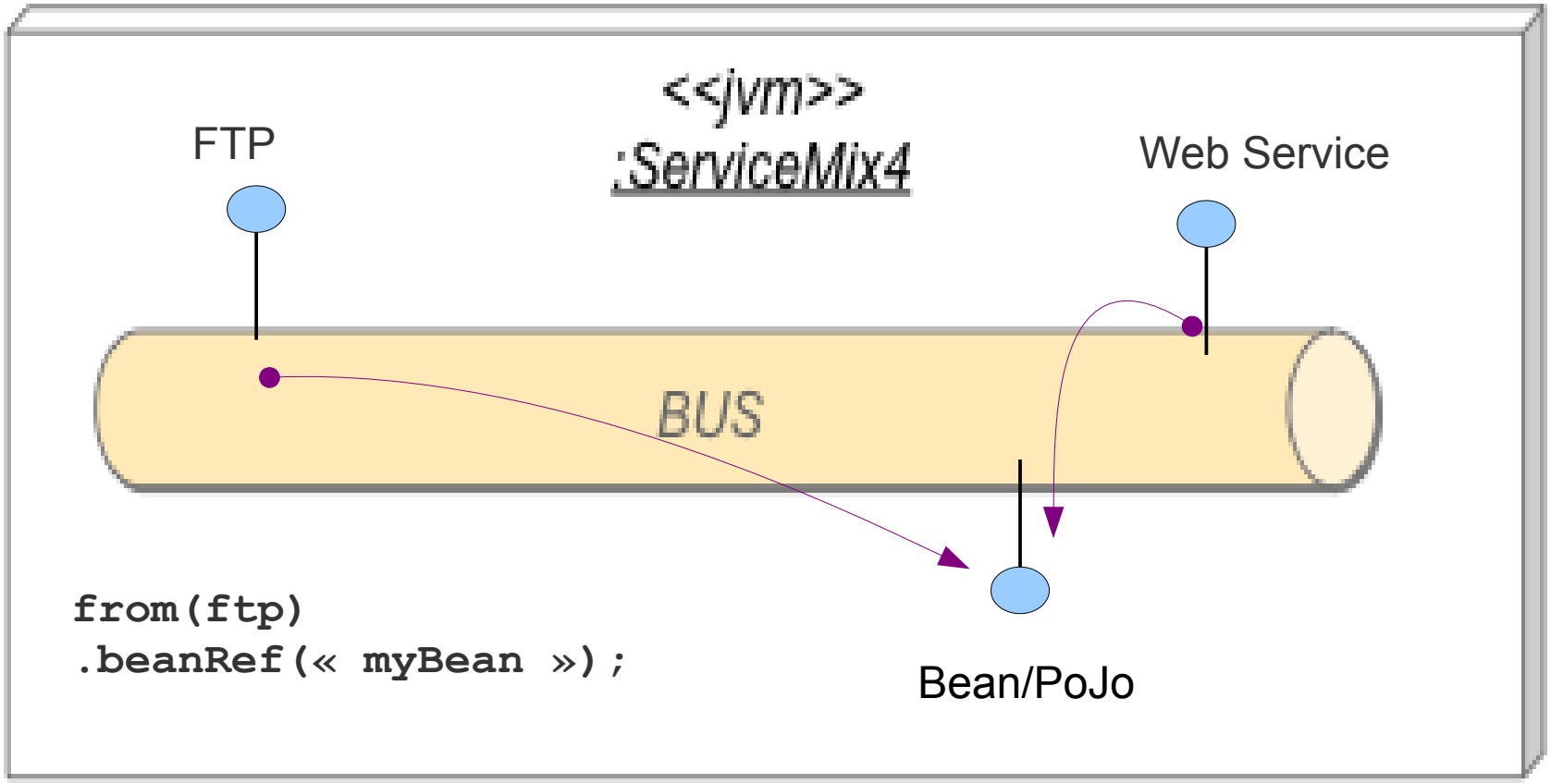
- Assistance de l' IDE



« Quelles sont les architectures possibles – messages, java et/ou web »

Architecture - messaging

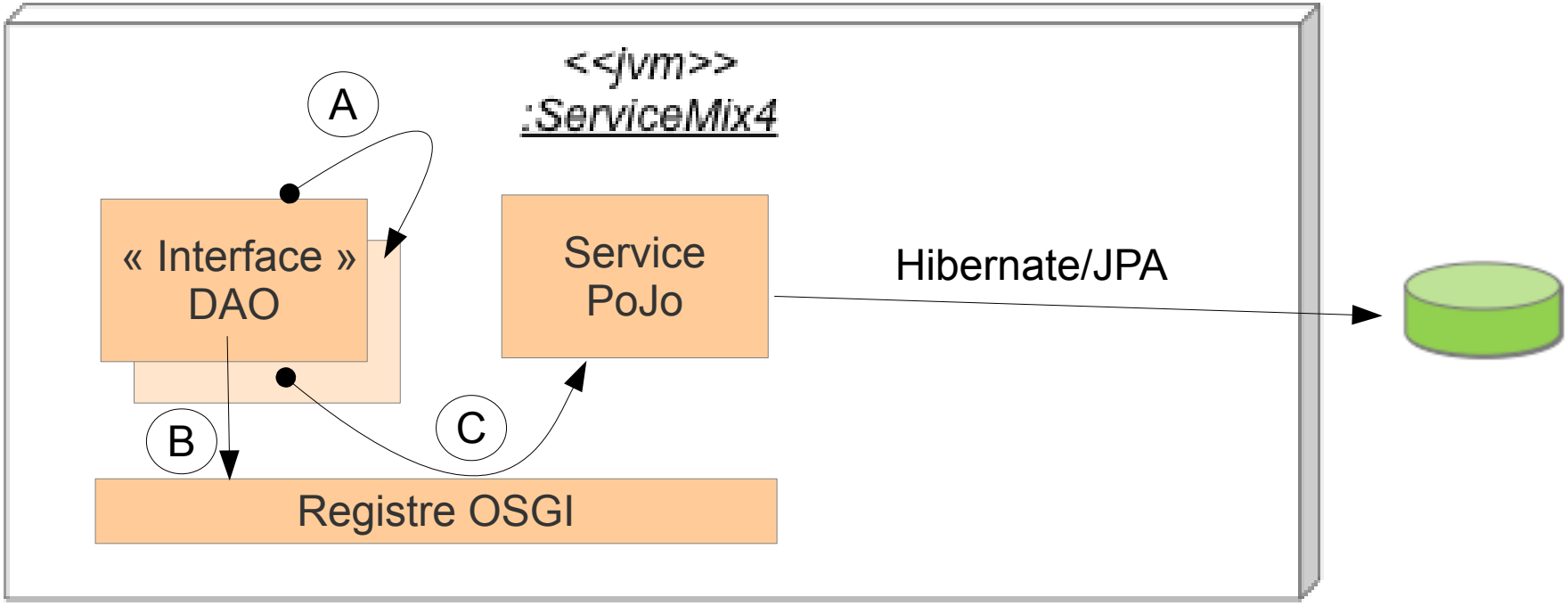
- Le BUS orchestre les échanges entre les endpoints




```
from(« cxf:bean:myWS »)
.beanRef(« myBean ») ;
```

Architecture - Java

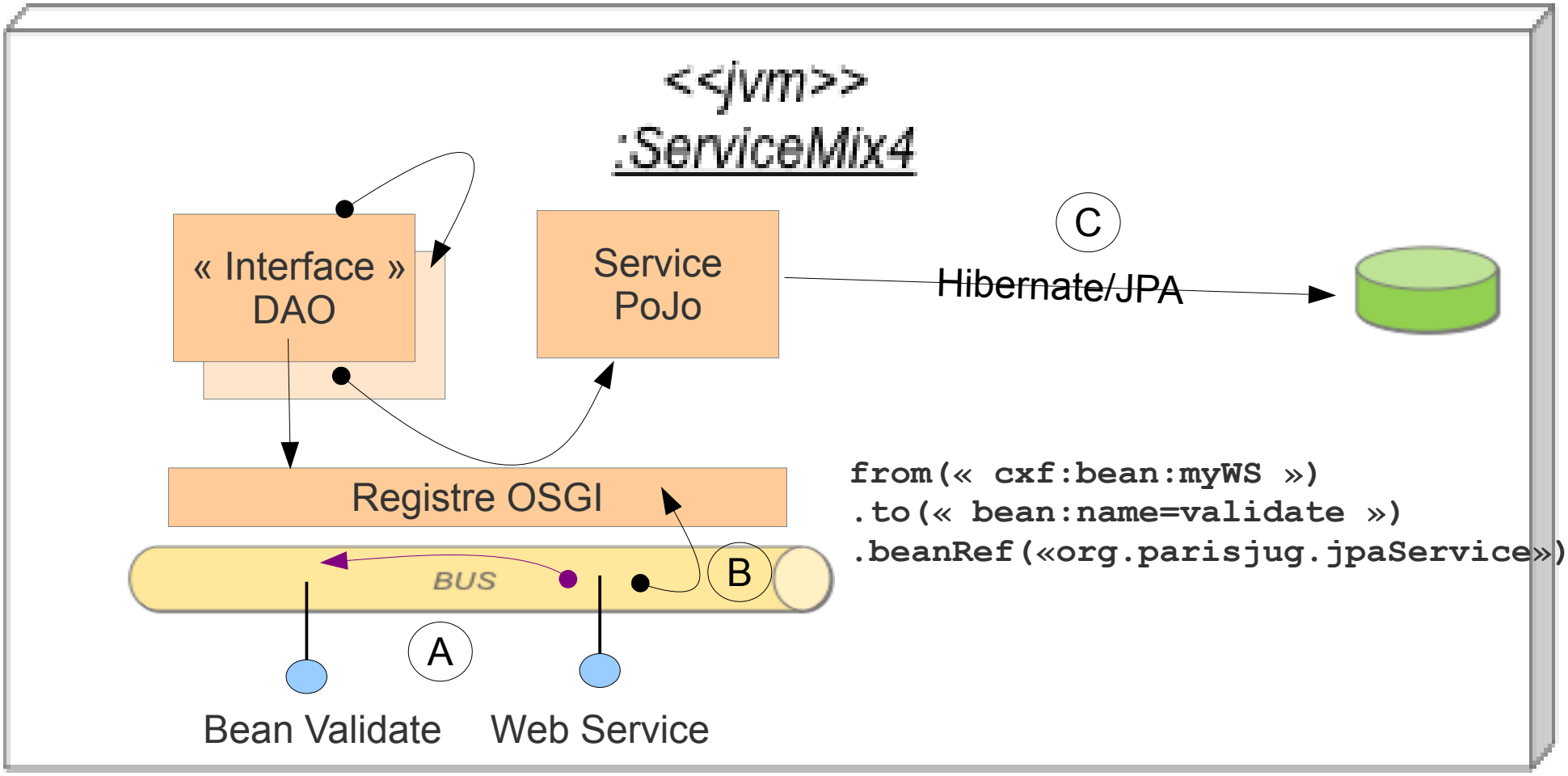
- La plateforme OSGI sert de registre pour des services = “Interface”




```
<bean id="incidentServiceTarget" class="org.apache.camel.service.impl.IncidentServiceImpl">
    <property name="incidentDAO">
        <osgi:reference interface="org.apache.camel.dao.IncidentDAO"/>
     </property>
</bean>
```

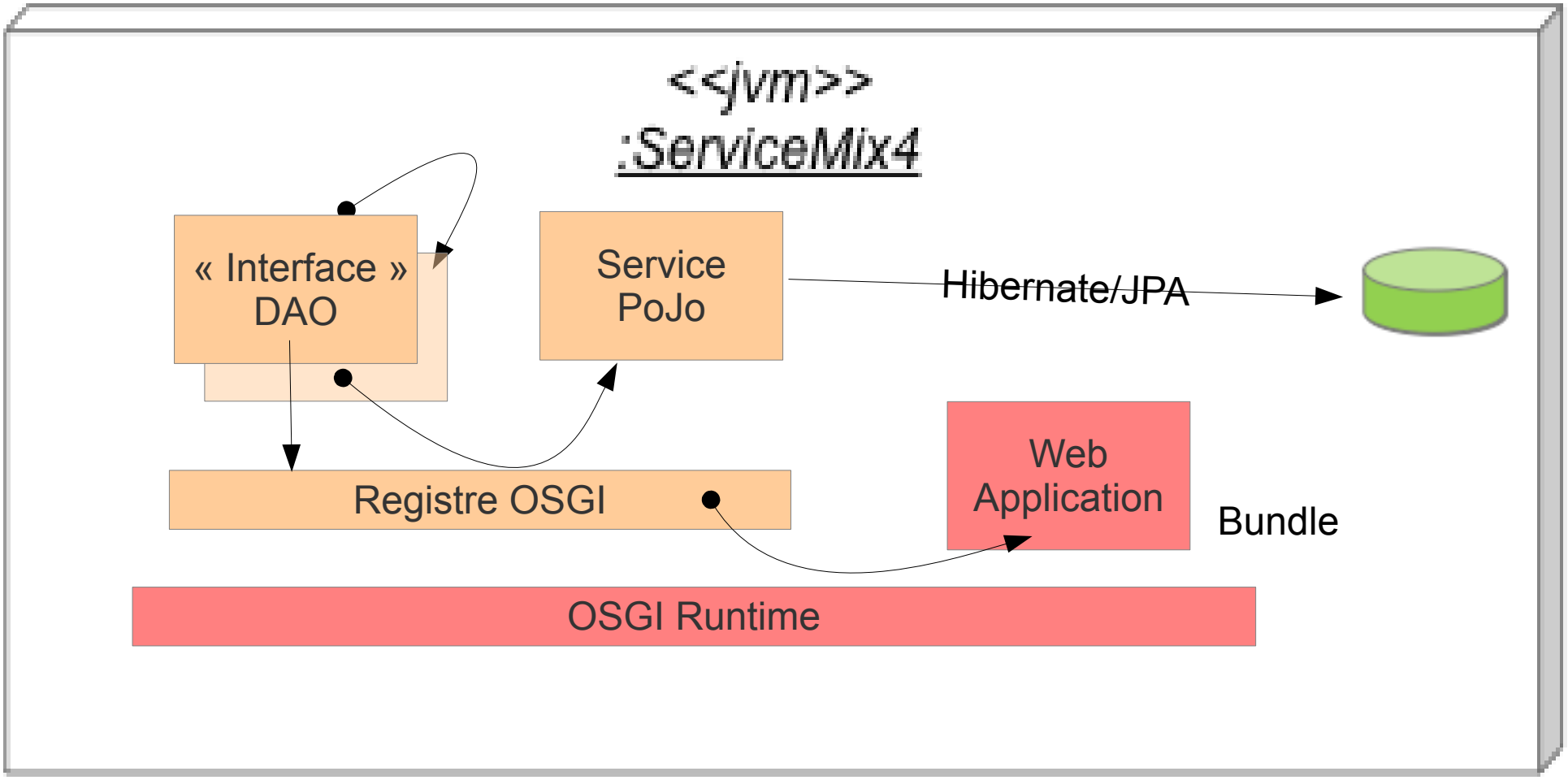
Architecture – Messaging + Java

- Decouplage des composants « services » de la couche d'intégration



Architecture – Java + Web

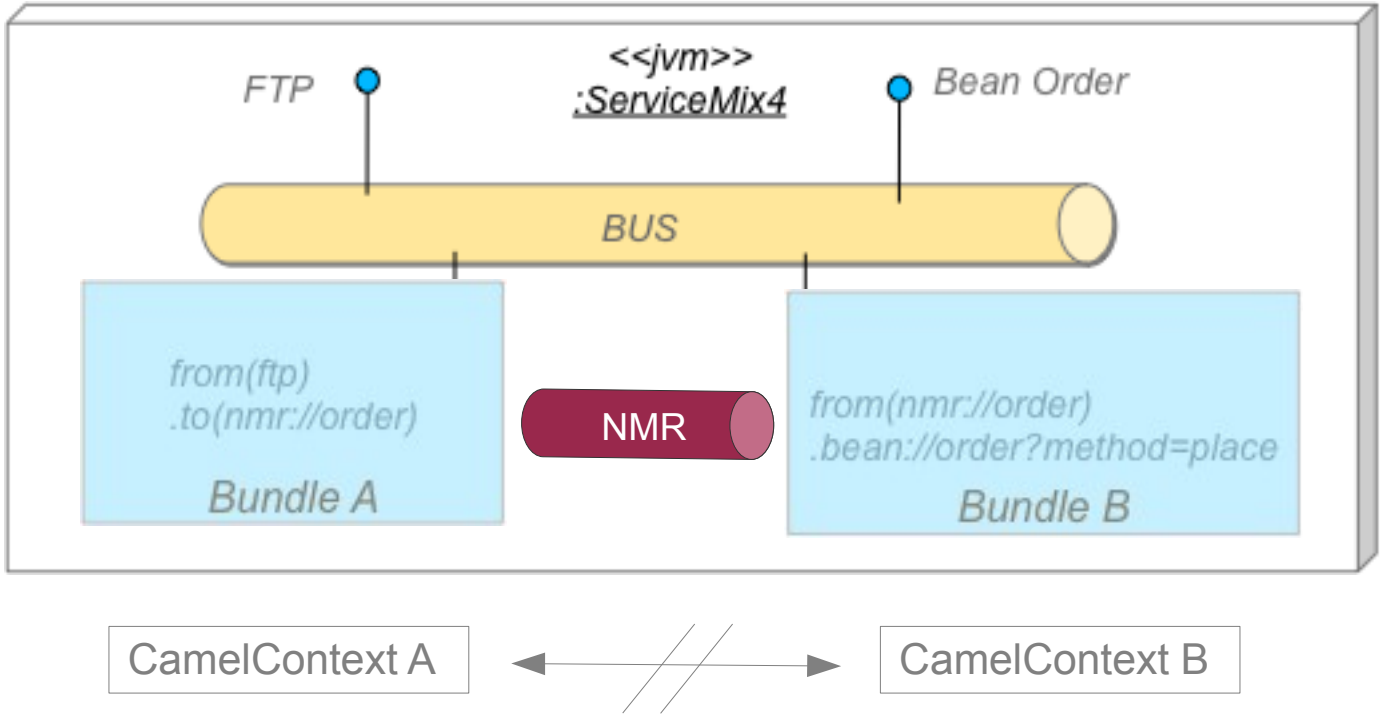
- L' ESB héberge les projets Web (Jetty + Pax Web + OSGI HTTP Service)



« Haute disponibilité, scalabilité et clustering »

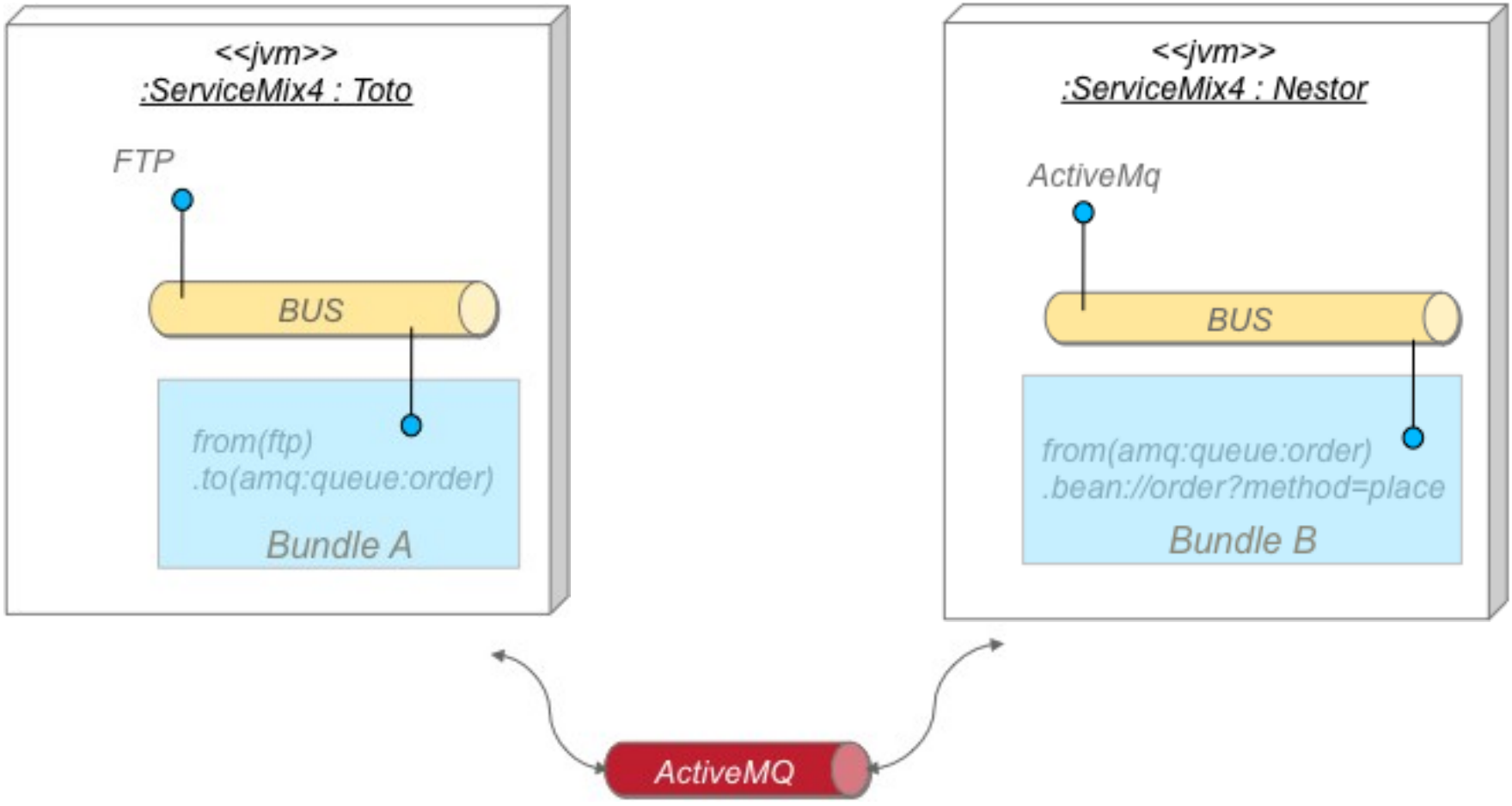
Une seule instance

- Unité de déploiement = bundle
- 1 seule instance SMX, plusieurs routes, cross-communication via NMR, asymétrie



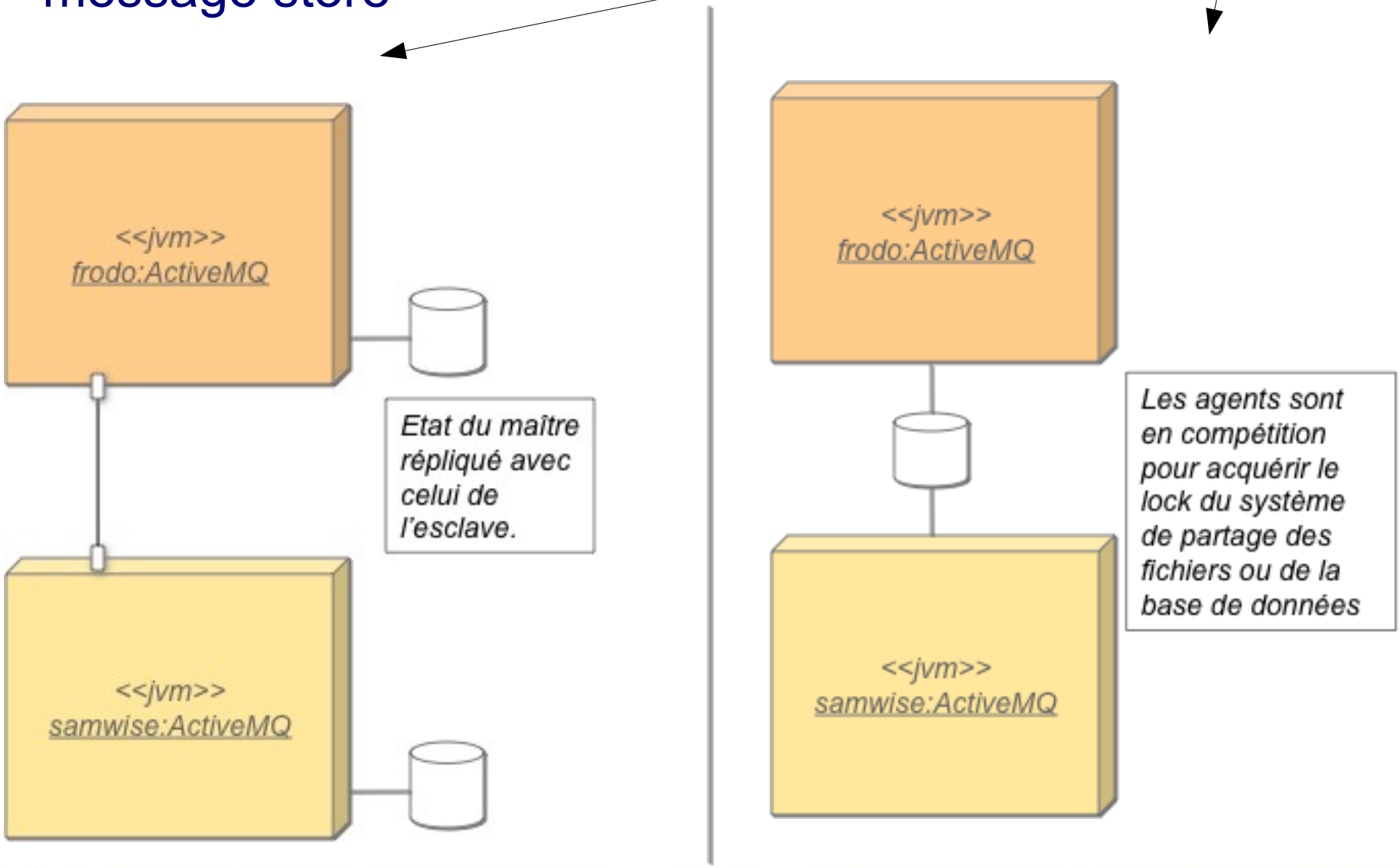
Plusieurs instances

- Bridge assuré par ActiveMQ
- “Interconnecte” les différentes instances de SMX



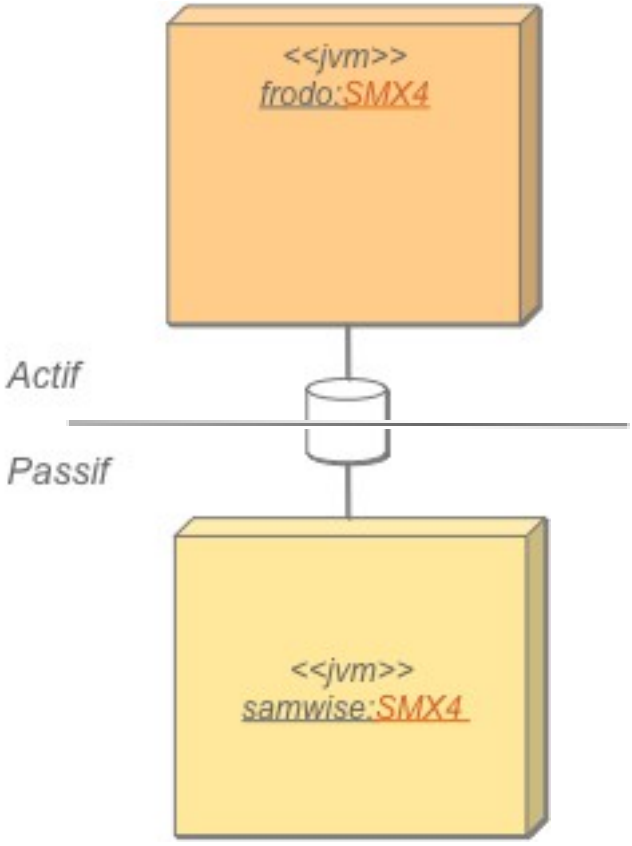
Haute disponibilité - ActiveMQ

- 2 mécanismes disponibles : pure Master/Slave et Shared message store



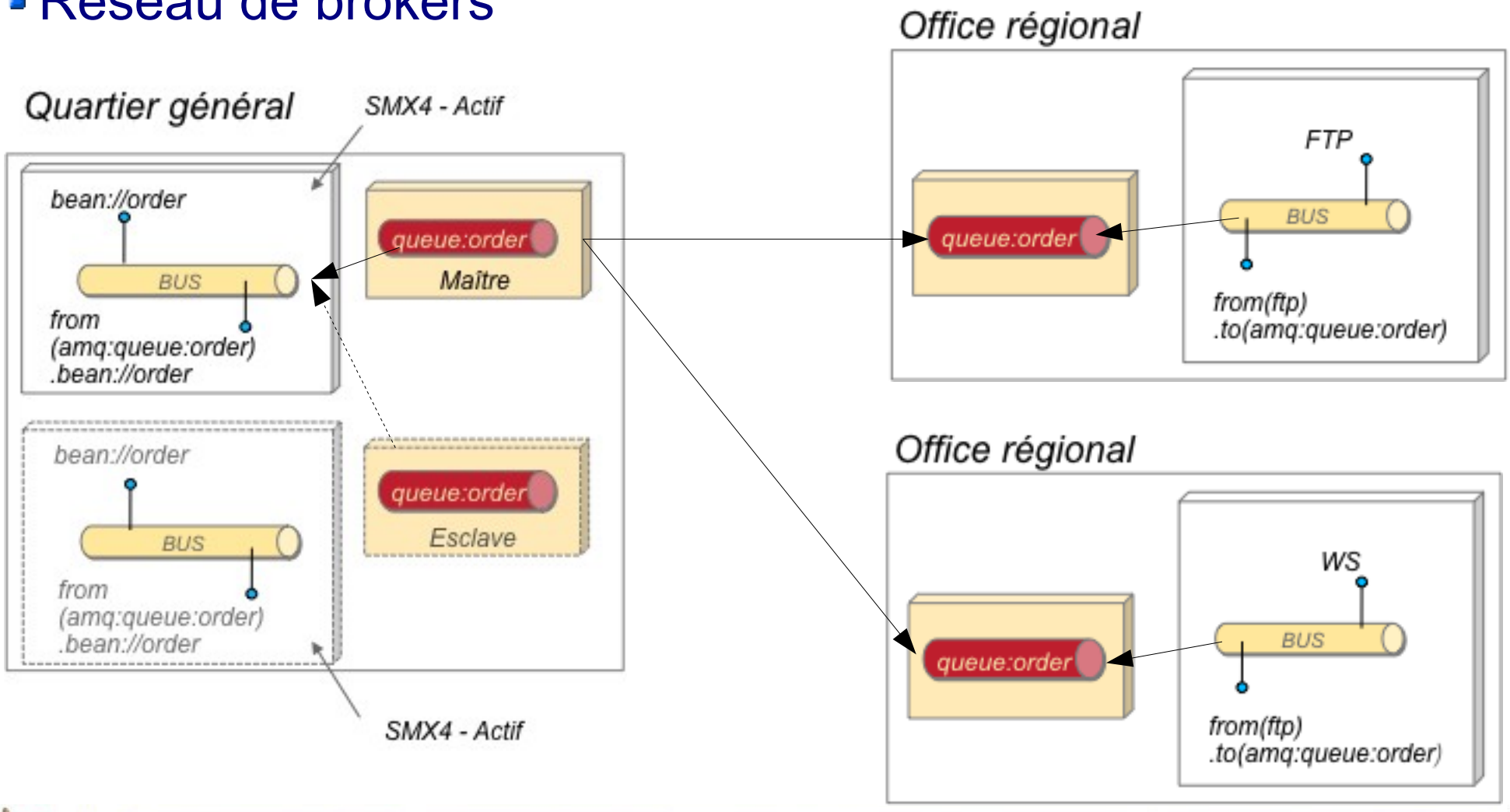
Haute disponibilité - ServiceMix

- Seul le mécanisme de (lock) shared store est possible
- Bundles peuvent etre deployés mais non activés



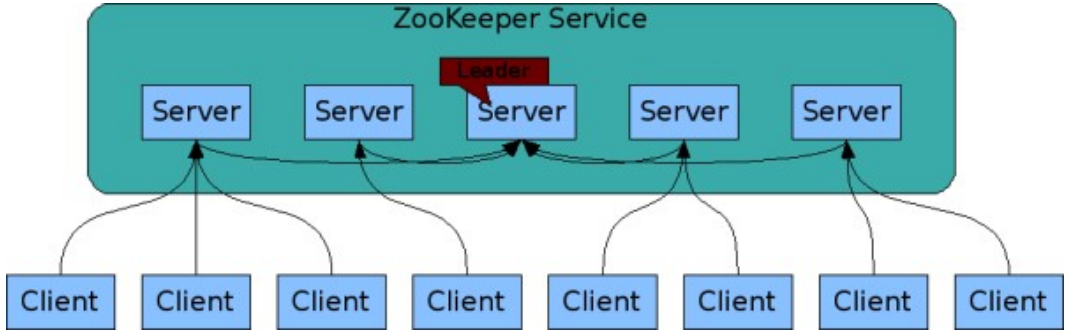
Haute disponibilité - Clustering

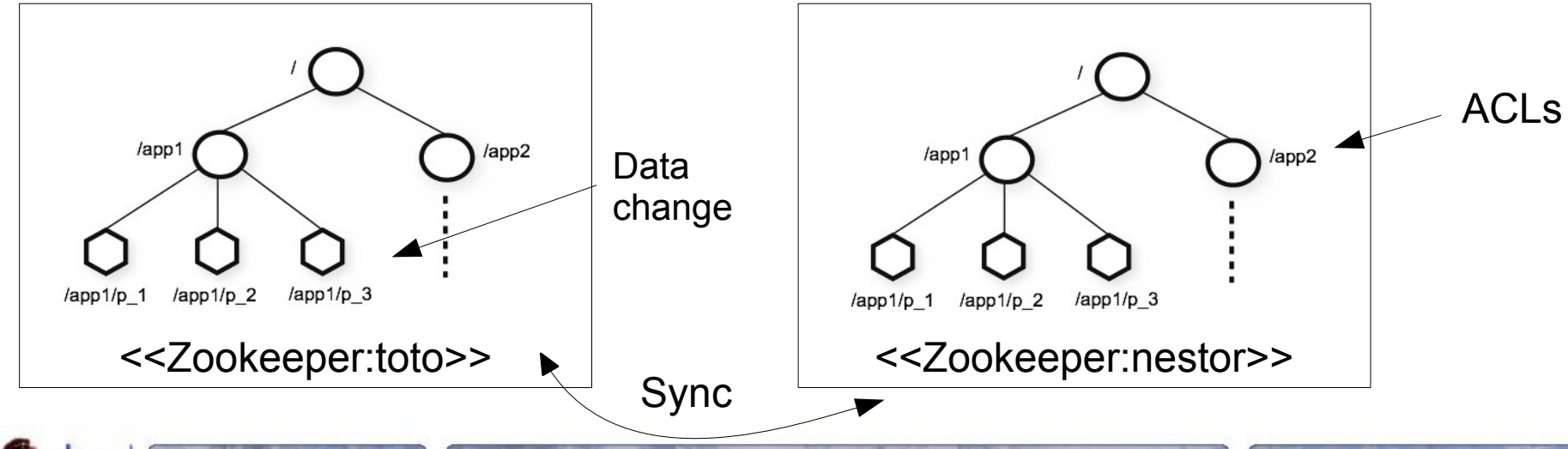
- Combine ActiveMQ (shared store) et ServiceMix (shared lock)
- Réseau de brokers



Cloud, Provisionnement

- FuseSource Fabric - basé sur Apache Zookeeper
- Service de coordination distribué hautement disponible utilisant un registre utilisé pour les configs du cluster et les noeuds (runtime)





Cloud, Provisionnement

- Mécanisme de provisionnement et de déploiement à distance
- Basé sur un profile associé à un agent

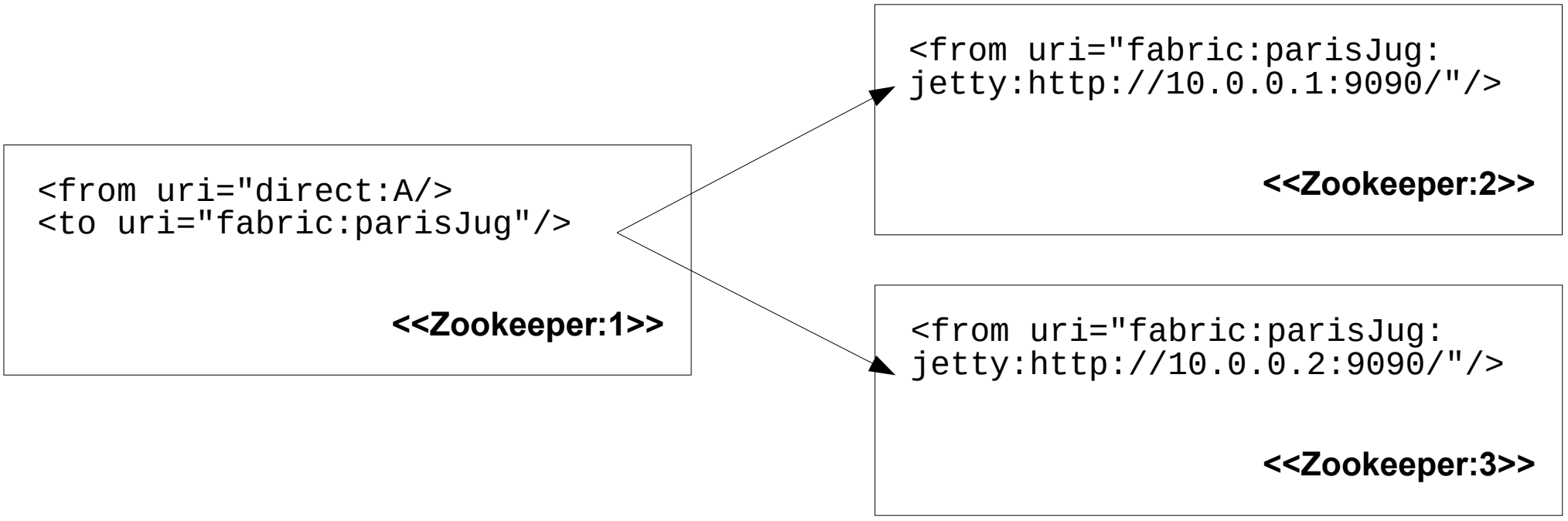
```
karaf@root> fabric:create-profile --parents default webserver
```

```
karaf@root> zk:create -r
/fabric/configs/versions/base/profiles/webserver/org.fusesource.fabric.agent/rep
ository.karaf mvn:org.apache.karaf.assemblies.features/standard/2.2.1-
SNAPSHOT/xml/features
karaf@root> zk:create -r
/fabric/configs/versions/base/profiles/webserver/org.fusesource.fabric.agent/fea
ture.war war
```

```
karaf@root> fabric:create-agent --profile webserver --parent root test
```

Cloud, Provisionnement

- La fabric permet d' enregister logiquement un endpoint camel dans le registre de Zookeeper (=znode)
- Offre une nouvelle stratégie pour les architectures HTTP, CXF pour balancer les requetes



« Fin de la premiére partie »

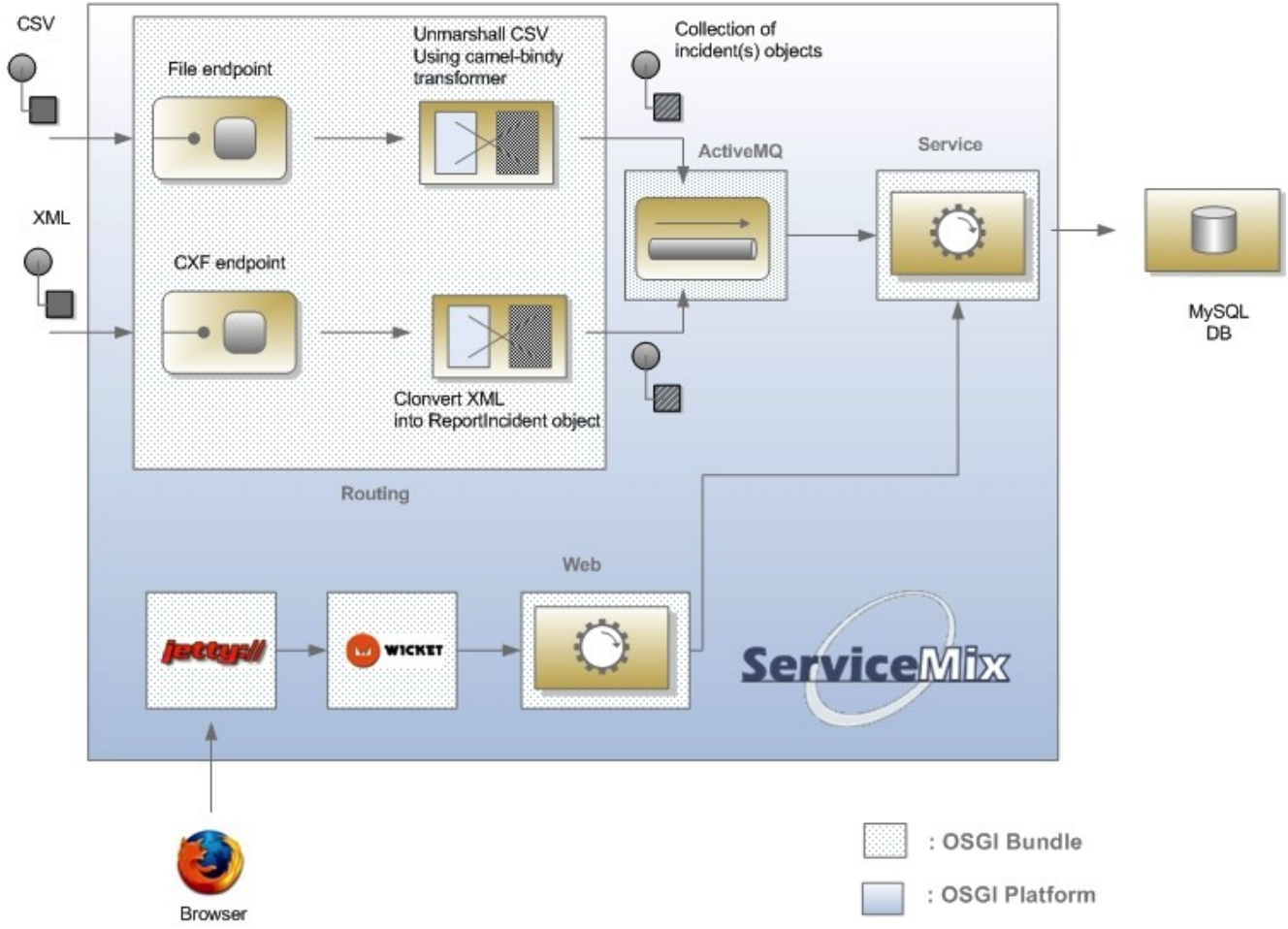
www.parisjug.org

Démonstration

Sommaire

- De la théorie à la pratique
- Transposition vers le language camel DSL des EIP
- Codage, déboggage, testing
- Développement des routes Camel
- Packaging et déploiement

Contexte



http://camel.apache.org/tutorial-osgi-camel-part2.html

Conclusion

Questions ?

Suivez-moi sur

 http://twitter.com/cmoulliard

 http://www.linkedin.com/in/charlesmoulliard

 Mon blog : http://cmoulliard.blogspot.com

Bibliographie / liens

- Integration - Camel : http://camel.apache.org
- ESB - ServiceMix : http://servicemix.apache.org
- Runtime - Karaf : http://karaf.apache.org
- WebService - CXF : http://cxf.apache.org
- Broker - ActiveMQ : http://activemq.apache.org
- OSGI EE - Aries : http://aries.apache.org
- FuseSource : http://fusesource.com/documentation/
- EIP : http://www.enterpriseintegrationpatterns.com

Sponsors