Generalitat de Catalunya
Centre de Telecomunicacions
i Tecnologies de la Informació

Framework Corporatiu J2EE

Servei d'Integració amb Cues (JMS)

Versió 1.0

Barcelona, 21 / febrer / 2006

Històric de modificacions

Data	Autor	Comentaris	Versió
16/01/2006	Atos Origin, sae openTrends	Versió inicial del document	1.0

Llegenda de Marcadors

Índex

1. Introducció

1.1. Propósit

Aquest servei permet configurar i usar de forma senzilla la infraestructura de missatgeria estándard JMS (Java Messaging Service) de J2EE . Aquest estàndar defineix dues modalitats:

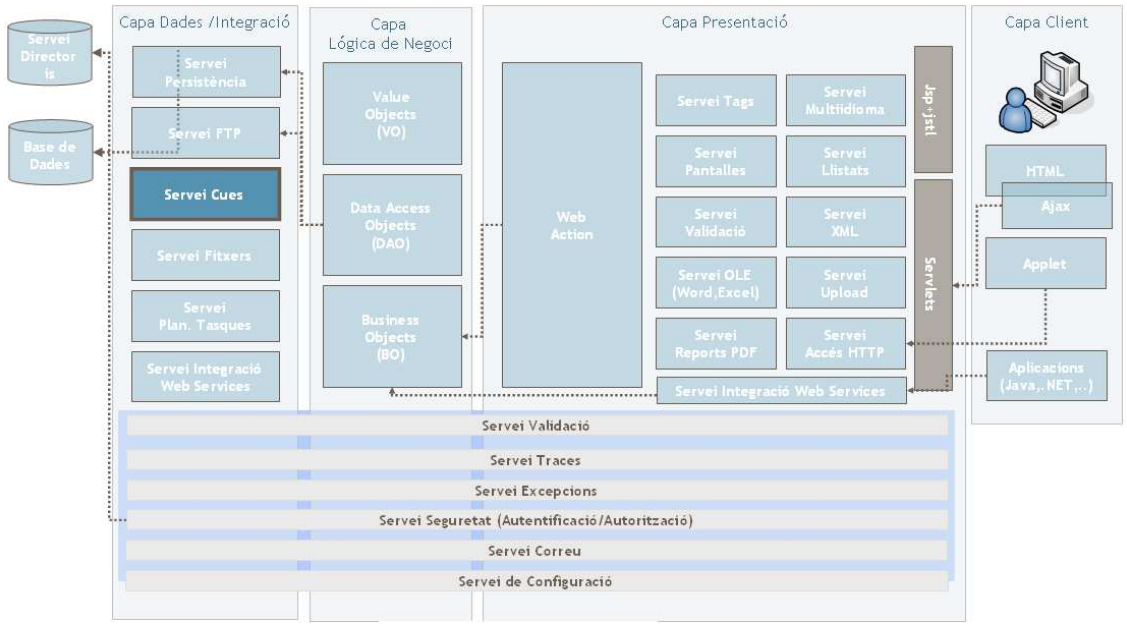
- Producció de missatges
- Consum de missatges, que pot ser asíncron.

El consum de missatges assíncrons a l'estàndar J2EE es realitza mitjançant "Message-Driven Beans". openFrame no s'incorpora cap mecanisme específic pel consum de missatges assíncrons, i la seva disponibilitat dependrà de la plataforma (especificament del Servidor d'aplicacions).

Així doncs, l'enfoc del servei és el de simplificar la publicació de missatges i la definició i configuració de destinataris.

1.2. Context i Escenaris d'Ús

El Servei d'Integració de Cues JMS es troba ubicat dins els serveis continguts a la capa de Dades/Integració de openFrame.



1.3. Versions i Dependències

En el present apartat es mostren quines són les versions i dependències necessàries per fer ús del Servei.
Dins la llista de dependències es mostren diferenciades:

- Dependències bàsiques. Llibreries necessàries per fer ús del servei
- Dependències adicionals. Aquestes dependències són necessàries per poder fer ús de característiques concretes del servei o per l'ús dels tests unitaris proporcionats amb el servei

1.3.1. Dependències Bàsiques

Nom	Tipus	Versió	Descripció
commons-beanutils-core	jar	1.7.0	
commons-codec	jar	1.3	
commons-digester	jar	1.5	
commons-logging	jar	1.0.4	http://jakarta.apache.org/commons
commons-logging-api	jar	1.0.4	
concurrent	jar	1.3.4	
jms	jar	1.1	
log4j	jar	1.2.9	
openFrame-services-configuration	jar	1.0	
openFrame-services-exceptions	jar	1.0	
openFrame-services-logging	jar	1.0	
spice-jndikit	jar	1.1	
spring	jar	1.2.5	

1.3.2. Dependències Adicionals

- OpenJms
  Dependències necesàries en cas de fer ús del Servidor de Cues OpenJMS

Nom	Tipus	Versió	Descripció
exolabcore	jar	0.3.7	
openjms	jar	0.7.7-alpha-1	
openjms-common	jar	0.7.7-alpha-1	
openjms-net	jar	0.7.7-alpha-1	

1.4. A qui va dirigit

Aquest document va dirigit als següents perfils:

- Programador. Per conéixer l'ús del servei
- Arquitecte. Per conéixer quins són els components i la configuració del servei
- Administrador. Per conéixer cóm configurar el servei en cadascun dels entorns en cas de necessitat

1.5. Documents i Fonts de Referència

[1] Spring JMS http://static.springframework.org/spring/docs/1.2.x/reference/jms.html

1.6. Glossari

2. Descripció Detallada

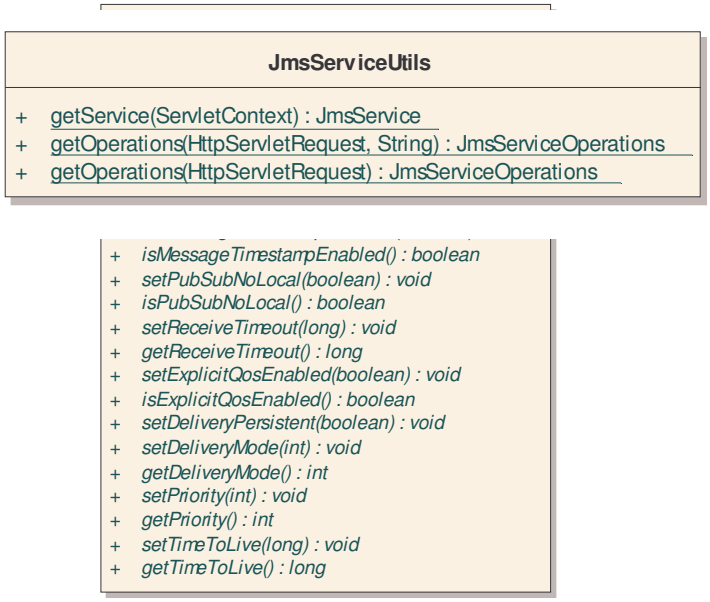
2.1. Arquitectura i Components

Els components podem classificar-los en:

- Interfícies i Components Genèrics. Interfícies del servei i components d'ús general amb independència de la implementació escollida.
- Implementació basada en Spring

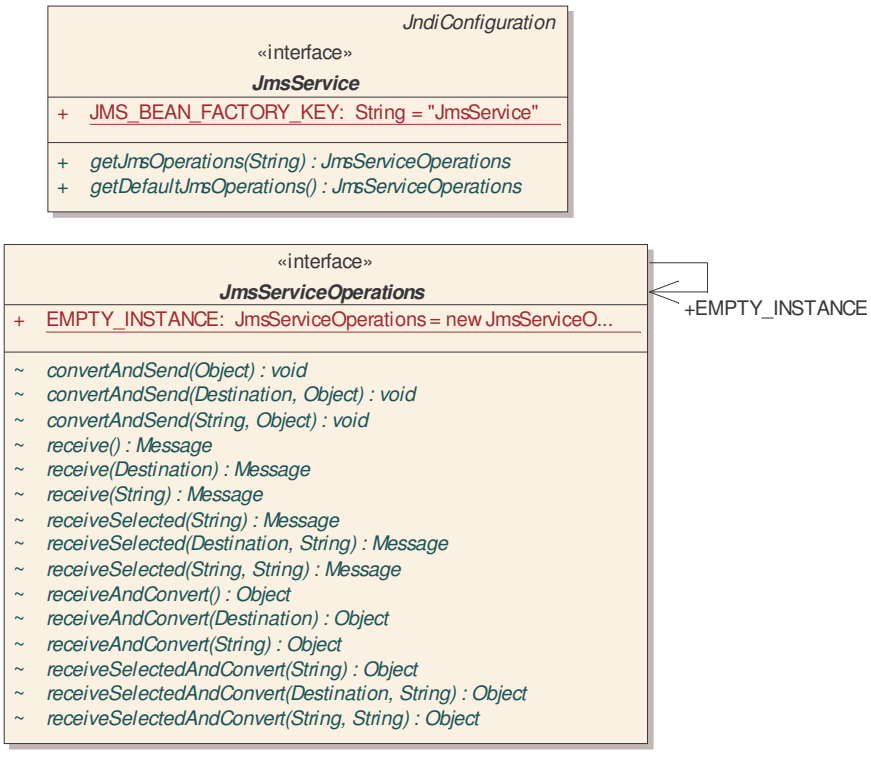
2.1.1. Interfícies i Components Genèrics

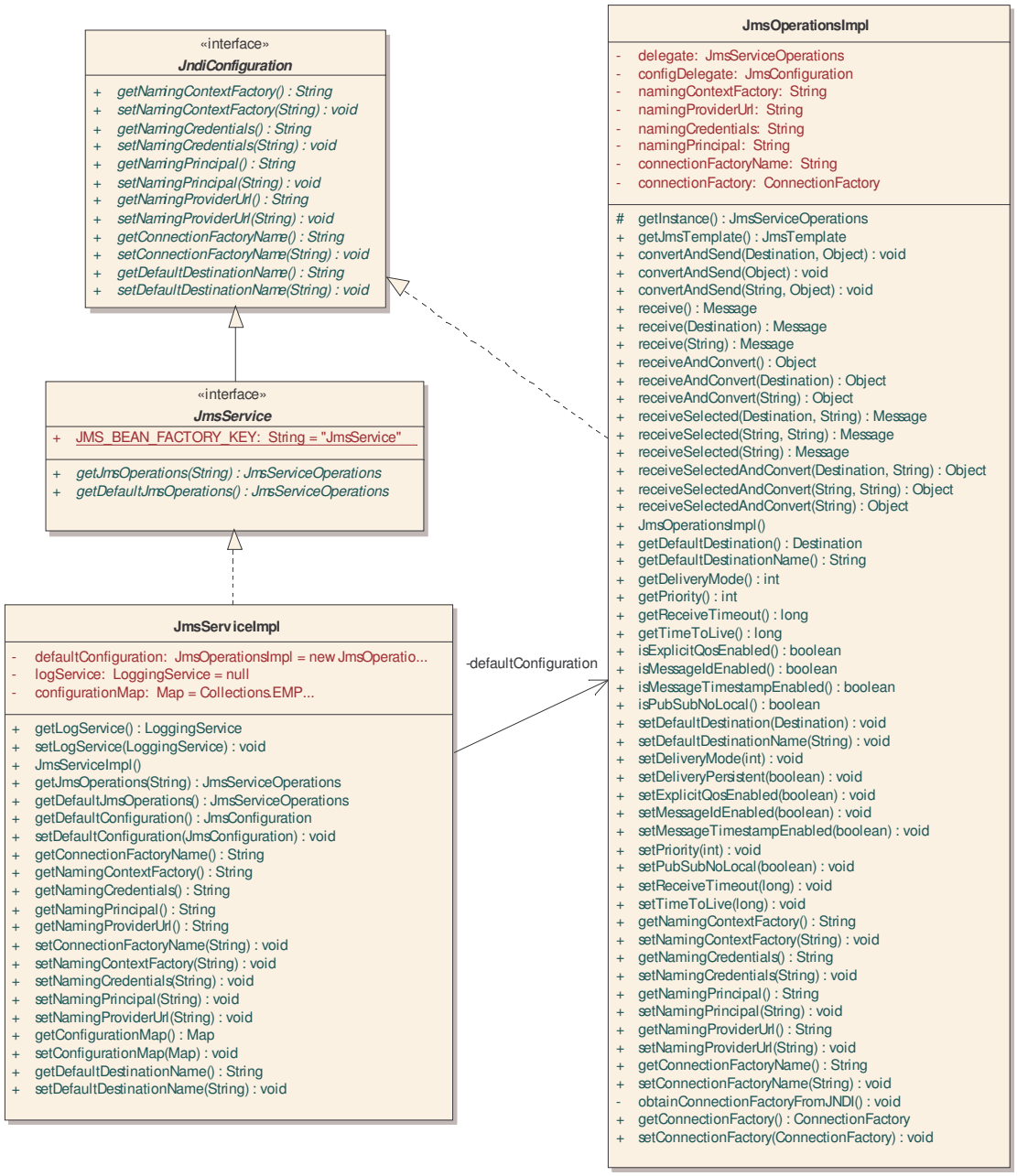
Component	Package	Descripció
JmsService	net.opentrends.openframe.services.jms	Interfície del servei que permet obtener la interfície JmsServiceOperations
JmsServiceOperations	net.opentrends.openframe.services.jms	Interfície que permet l'enviament i recepció de missatges
JmsConfiguration	net.opentrends.openframe.services.jms	Configuració del servei
JmsServiceException	net.opentrends.openframe.services.jms.exception	Excepció llençada pel servei
JmsServiceUtils	net.opentrends.openframe.services.jms	Classe que permet obtenir el servei des de classes Action



2.1.2. Implementació basada en Spring

Component	Package	Descripció
JndiConfiguration	net.opentrends.openframe.services.jms.impl	Interfície d'accés a la font JNDI (per accedir a la definició JMS feta al servidor)
JmsServiceImpl	net.opentrends.openframe.services.jms.impl	Implementació de la interfície 'JmsService'
JmsOperationsImpl	net.opentrends.openframe.services.jms.impl	Implementació de la interfície 'JmsServiceOperations' que fa ús de classes específiques de Spring





2.2. Instal·lació i Configuració

2.2.1. Instal·lació

La instal·lació del servei requereix de la utilització de la llibreria 'openFrame-services-jms' i les dependències indicades a l'apartat 'Introducció-Versions i Dependències'.

2.2.2. Configuració

La configuració del servei implica els següents pasos:

1) Definir la implementació del Servei que s'usarà
2) Definir la localització del fitxer de propietats del Servei
3) Definir les propietats del Servei

Definició del Servei

```
<bean id="webServicesService"
      ...>
```

Fitxer de configuració: openFrame-services-jms.xml

Ubicació proposada: <PROJECT_ROOT>/src/main/resources/spring

En aquest apartat configurarem el bean de Servei de Cues, és a dir, la implementació que es farà servir en l'atribut 'class'. En l'actualitat s'ofereix la classe ' net.opentrends.openframe.services.jms.impl.JmsServiceImpl'

```
   <bean id="JmsService"
class="net.opentrends.openframe.services.jms.impl.JmsServiceImpl">
...
   </bean>
```

Podem definir les següents propietats:

Propietat	Requerit	Descripció
logService	No	Referència al Servei de Traces
namingProvider Url	Sí	La URL del servei JNDI (protocol TCP o RMI, generalment), que permetrà localitzar el servei JMS remot
namingContextF actory	Sí	La implementació de la factoriia JNDI que permet localitzar el servei. Exemple: Per OpenJMS s'ha d'especificar "org.exolab.jms.jndi.InitialContextFactory"
connectionFacto ryName	Sí	Nom JNDI amb el que es troba registrada la factoria de connexions JMS
defaultDestinatio nName	Sí	Nom del destí (cua o tópic) en el servidor JMS remot
namingPrincipal	Sí	Usuari
namingCredentia ls	Sí	Password

Per les propietats es recomana que siguin definides de forma externa a un fitxer de propietats i referenciades aquí amb el format ${nom} (veure 'Definir les propietats del Servei').

En cas de que volguessim fer ús de diferents destins de publicació es farà ús de la propietat 'configurationMap', on la clau correspondrà al nom de la configuració (accessible després amb el mètode 'getJmsOperations(nom)') i podrem especificar les propietats indicades a dalt pel valor. Exemple:

```
<bean id="JmsService"
          class="net.opentrends.openframe.services.jms.impl.JmsServiceImpl">
...
   <property name="configurationMap">
     <map>
       <entry key="otherConfig">
        <bean parent="baseJmsConfigurator">
          <property name="defaultDestinationName" value="topic2"/>
          <property name="namingContextFactory"
          value="org.exolab.jms.jndi.InitialContextFactory"/>
          <property name="namingProviderUrl"
          value="rmi://localhost:1099"/>
          <property name="namingPrincipal" value="admin"/>
          <property name="namingCredentials" value="openjms"/>
          <property name="connectionFactoryName"
          value="ConnectionFactory"/>
         </bean>
        </entry>
     </map>
   </bean>
```

Definició de la Localització del Fitxer de Propietats del Servei

```
<bean id="configManager"
      ...>
```

Fitxer de configuració: openFrame-services-configuration.xml

Ubicació proposada: <PROJECT_ROOT>/src/main/resources/spring

Seguint el propósit general del Servei de Configuració definirem a la propietat 'basePropertyFiles' una nova localització pel fitxer de propietats del servei.

Exemple:

```
<bean id="configurationService"
class="net.opentrends.openframe.services.configuration.springframework.beans.facto
ry.config.HostPropertyPlaceholderConfigurer">
<property name="basePropertyFiles">
   <list>
          ...
          <value>classpath:jms/jms.properties</value>
   </list>
   </property>
</bean>
```

A l'exemple, s'ha definit la ubicació del fitxer a 'classpath:jms/jms.properties' (veure següent pas).

Definició de les Propietats del Servei

Fitxer de configuració: jms.properties

Ubicació proposada: <PROJECT_ROOT>/src/main/resources/jms

En aquest fitxer definirem les propietats que permet el servei (veure 'Definició del Servei')
Exemple:

```
jmsService.namingProviderUrl=rmi://localhost:1099
jmsService.namingContextFactory=org.exolab.jms.jndi.InitialContextFactory
jmsService.connectionFactoryName=ConnectionFactory
jmsService.defaultDestinationName=topic1
jmsService.namingPrincipal=admin
jmsService.namingCredentials=openjms
```

En aquest cas, la referència en la definició del servei seria (només es mostra la referència a la primera propietat definida):

```
<property name="namingProviderUrl" value="${jmsService.namingProviderUrl}"/>
...
```

2.3. Utilització del Servei

S'han de seguir els següents pasos:

1) Obtenir el destí JMS

El mètode "getDefaultJmsOperations" permet obtenir el destí JMS per defecte que es configura junt amb el servei, mentre que "getJmsOperations" permet obtenir una configuració diferent, identificada per un nom.

2) Una vegada tenim la interfície 'JmsServiceOperations' podem realitzar les operacions que s'especifiquen a la seva interfície.

```
public interface JmsServiceOperations {

   void convertAndSend(Object message) throws JmsServiceException;
   void convertAndSend(Destination destination, Object message)
                  throws JmsServiceException;
   void convertAndSend(String destinationName, Object message)
                  throws JmsServiceException;
   Message receive() throws JmsServiceException;
```

```
    Message receive(Destination destination) throws
    JmsServiceException;
    Message receive(String destinationName) throws
    JmsServiceException;
    Message receiveSelected(String messageSelector) throws JmsServiceException;
    Message receiveSelected(Destination destination, String messageSelector)
                    throws JmsServiceException;
    Message receiveSelected(String destinationName, String messageSelector)
                    throws JmsServiceException;
    Object receiveAndConvert() throws JmsServiceException;
    Object receiveAndConvert(Destination destination)
                    throws JmsServiceException;
    Object receiveAndConvert(String destinationName) throws JmsServiceException;
    Object receiveSelectedAndConvert(String messageSelector)
                    throws JmsServiceException;
    Object receiveSelectedAndConvert(Destination destination,
                    String messageSelector) throws JmsServiceException;
    Object receiveSelectedAndConvert(String destinationName,
                    String messageSelector) throws JmsServiceException;

}
```

Per conéixer com usar de cadascuna de les operacions es requereix un coneixement previ de JMS.

2.4. Eines de Suport

2.4.1. OpenJMS

En cas de no disposar d'un Servidor de Cues per a realitzar les proves, es pot fer ús de OpenJMS. Accedir a 'http://openjms.sourceforge.net/' per a més referència.

2.5. Integració amb Altres Serveis

2.5.1. Integració amb el Servei Multiidioma i Excepcions

El servei llença vàries excepcions amb codis per tal que el missatge pugui ser internacionalitzat. En el següent exemple es mostren els codis que s'han de definir:

```
openFrame.services.jms.jndi_lookup_failed=Jndi lookup failed for {0}
```

2.6. Preguntes Freqüènts

3. Exemples

3.1. Exemple d'Enviament a una Destinació

3.1.1. Exemple de Prova Unitària

```
//Obtenim el context Spring de test, aixo  no cal fer-ho explícitament
//en una aplicació web

BeanFactory beanFactory = new
ClassPathXmlApplicationContext("applicationContext.xml");

//Obtenim el servei pròpiament dit

JmsServiceImpl JmsService =
   (JmsServiceImpl)beanFactory.getBean("JmsService");

 JmsServiceOperations operations = (JmsServiceOperations)

//ara es pot fer servir l'interface  JmsServiceOperations ...

 operations.convertAndSend("A String message to the default destination")
```

3.1.2. Exemple des d'una classe Action

```
    JmsServiceUtils.getOperations(request).convertAndSend("testMessage");
```

En aquest cas s'ha fet ús del mètode estàtic "getOperations" de la classe auxiliar "net.opentrends.openframe.services.jms.JmsServiceUtils"
Aquest mètode permet utilitzar el servei sense haver de localitzar-lo expressament.

4. Annexos