Generalitat de Catalunya
Centre de Telecomunicacions
i Tecnologies de la Informació

Framework Corporatiu J2EE

Servei de multiidioma

Versió 1.0

Barcelona, 21 / febrer / 2006

Històric de modificacions

Data	Autor	Comentaris	Versió
09/12/2005	Atos Origin, sae openTrends	Versió inicial del document	1.0

Llegenda de Marcadors

Índex

1. INTRODUCCIÓ 4

1.1. PROPÓSIT 4
1.2. CONTEXT I ESCENARIS D'ÚS 4
1.3. VERSIONS I DEPENDÈNCIES 5
1.4. A QUI VA DIRIGIT 5
1.5. DOCUMENTS I FONTS DE REFERÈNCIA 6
1.6. GLOSSARI 6

2. DESCRIPCIÓ DETALLADA 7

2.1. ARQUITECTURA I COMPONENTS 7
Arquitectura dels Filtres 7
Interfícies i Components Genérics 8
Components Implementació Spring 8
Components Integració amb la Capa de Presentació 9
2.2. INSTAL·LACIÓ I CONFIGURACIÓ 10
Instal·lació 10
Configuració 10
2.3. UTILITZACIÓ DEL SERVEI 15
Canviar l'Idioma de l'Usuari 15
Obtenció de Traduccions 15
2.4. INTEGRACIÓ AMB ALTRES SERVEIS 16
2.5. PREGUNTES FREQÜÈNTS 16

3. EXEMPLES 17

3.1. EXEMPLE DE TEST UNITARI 17

4. ANNEXOS 19

1. Introducció

1.1. Propósit

El Servei de multiidioma té com a objectiu principal permetre el desenvolupament d'aplicacions en el que no sigui necessària cap reenginyeria cada vegada que s'incorpori un nou llenguatge a l'aplicació.

En general tota aplicació internacionalitzada pot realitzar-se de 2 formes diferenciades:

- Mitjançant la rèplica d'un conjunt de pàgines per a cada idioma
- Mitjançant un únic conjunt de pàgines que obtenen diferents literals de forma externa segons l'idioma.

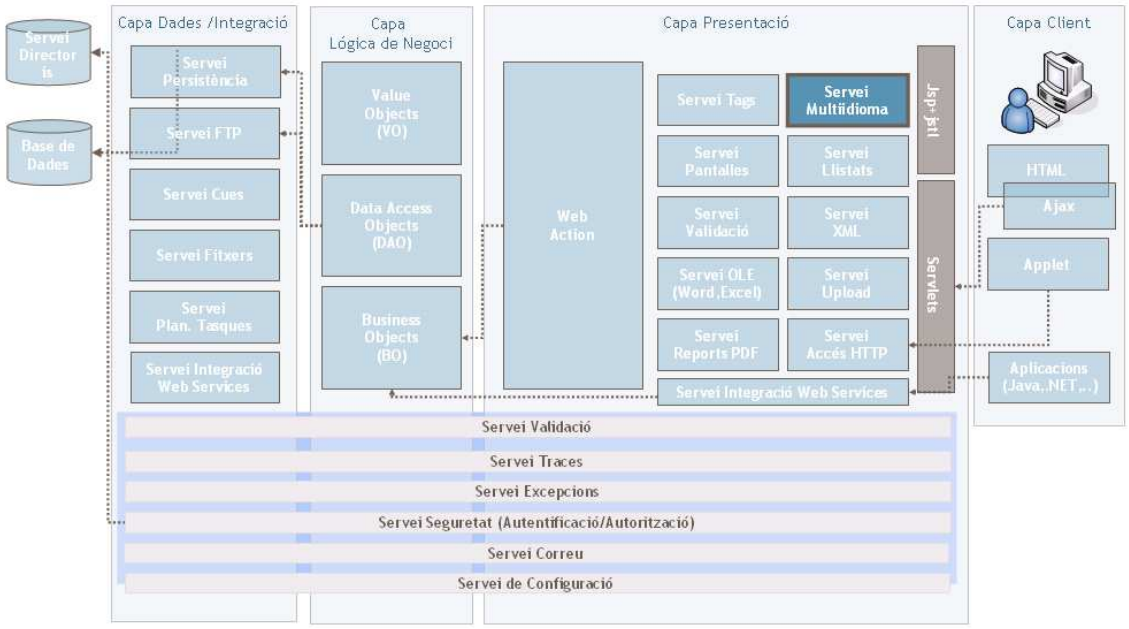
La primera solució, fins ara utilizada en moltes pàgines Web, implica un cost gran de rèplica i manteniment, ja que qualsevol nou requeriment o canvi comporta realitzar més d'un desenvolupament, un per cada idioma.

La segona solució, recomanada per openFrame permet que:

- Es puguin afegir llenguatges sense haver de realitzar canvis al codi.
- Els textes, imatges i missatges s'emmagatzemen de forma externa al codi.
- La informació (per exemple dates) es pot formatejar segons l'idioma de l'usuari

1.2. Context i Escenaris d'Ús

El Servei de Multiidioma es troba localitzat a la Capa de Presentació. El seu ús és necessari sempre i quan es requereixi de la internacionalització de les nostres aplicacions. En qualsevol cas encara que es faci ús d'un únic idioma a l'aplicació es recomana seguir la seva internacionalització per preveure futurs canvis i adaptacions.



El Servei de Multiidioma és utilitzat pel framework entre d'altres pel:

- Servei de Tags. Per generar els literals dels components d'entrada de formularis
- Arquitectura base de la Capa de Presentació. Per la integració transparent amb JSTL i Struts en l'ús de la internacionalització

1.3. Versions i Dependències

En el present apartat es mostren quines són les versions i dependències necessàries per fer ús del Servei.

Nom	Tipus	Versió	Descripció
openFrame-core	jar	1.0	
openFrame-services-exceptions	jar	1.0	
openFrame-services-logging	jar	1.0	
spring	jar	1.2.5	

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

1.6. Glossari

i18N (Internationalization)

i18N o Internationalization (el 18 correspon a les 18 lletres que hi ha entre la I inicial i la n final) és el procés d'agafar una aplicació dissenyada i reestructurar-la per a que pugui ser usada en diferents localitats o bé definir el procés de crear-la per a ser totalment flexible per executar-se en qualsevol localitat. Java defineix les classes bàsiques d'ús de la internacionalització.

2. Descripció Detallada

2.1. Arquitectura i Components

El Servei es basa en l'ús de la internacionalització i18N. Seguint el patró comú a tots els serveis defineix una interfície i ofereix implementacions per diferents tecnologies específiques.

Els components podem classificar-los en:

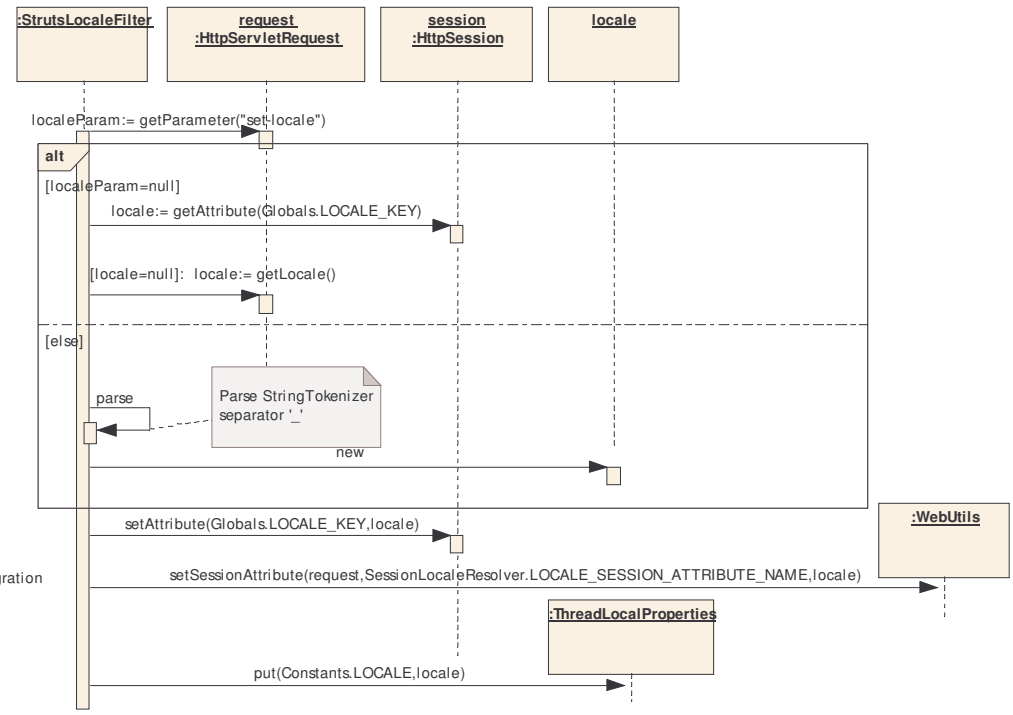
- Interfícies i Components Genérics. Interfícies del servei i components d'ús general amb independència de la implementació escollida.
- Implementació de les interfícies basada en Spring
- Components d'integració amb la capa de presentació

Arquitectura dels Filtres

La internacionalització té en consideració la configuració de l'idioma de l'usuari al seu navegador. D'aquesta forma, els literals mostrats s'obtindran del fitxer de traduccions associat a aquest idioma.

En qualsevol cas, és possible especificar un paràmetre 'set-locale' amb el llenguatge i el país com a valor separats per '_'. En el següent diagrama podem veure quin és el procediment de tractament de tota petició:

- Si no es rep el paràmetre 'set-locale', es considera si existeix la variable de Struts que especifica l'idioma de la sessió de l'usuari . Si aquest no està especificat, s'obté de la petició HTTP. Aquest idioma serà el que es troba configurat en primer lloc a la llista d'idiomes del navegador (dins 'Preferències').



- Si es rep el paràmetre 'set-locale', el seu valor és parsejat en 2 parts i es crea l'objecte 'Locale' corresponent

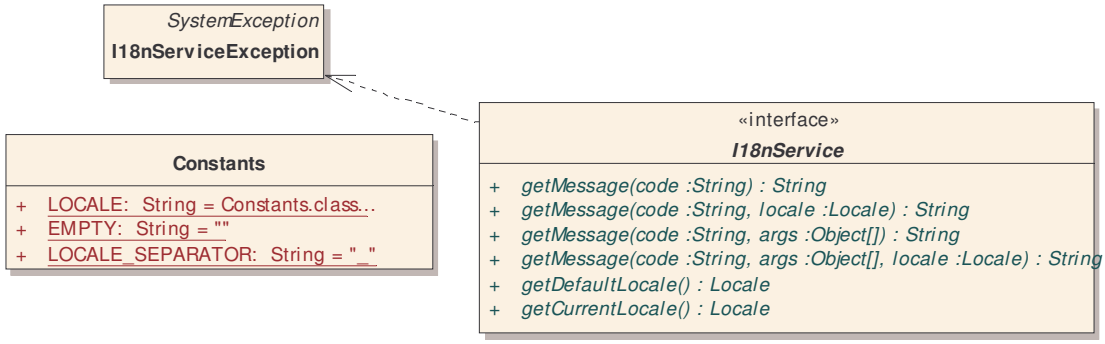
Finalment, amb el locale obtingut es realitzen 3 pasos:

1) Introducció del locale escollit a la sessió de Struts
2) Introducció del locale a la sessió de Spring
3) Introducció del locale a l'objecte 'ThreadLocalProperties' de openFrame. Aquest objecte permet passar informació entre els objectes dins el fil d'execució actual, i serà usat pel mètode 'getCurrentLocale' de la implementació del servei d'internacionalització.

Interfícies i Components Genérics

El servei de traces defineix les següents interfícies i components genérics:

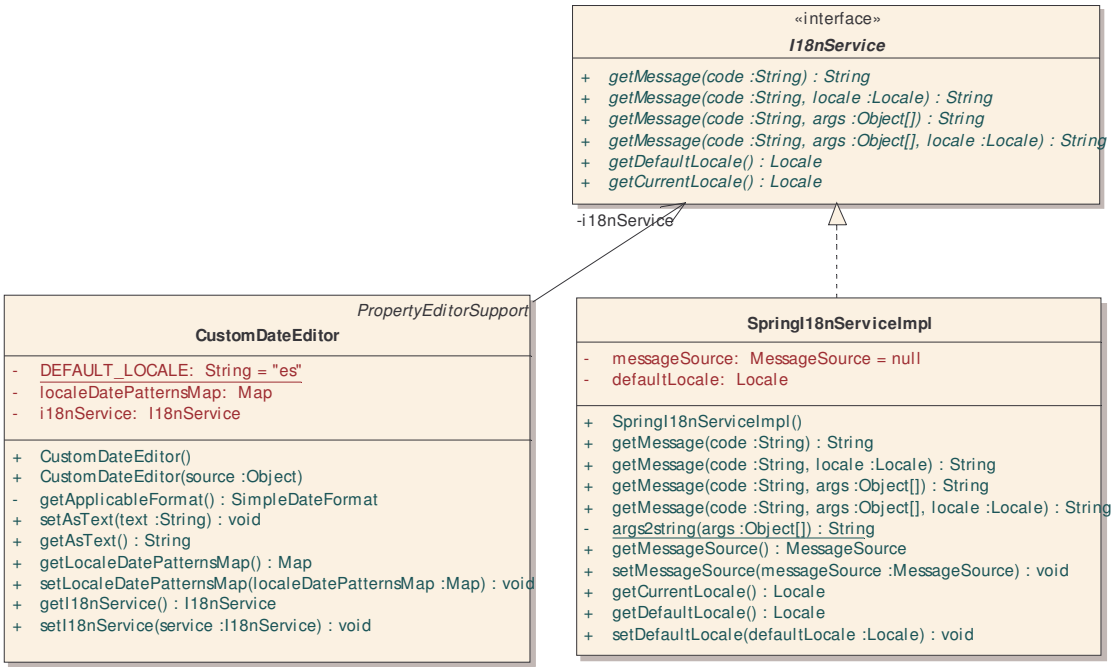
Component	Package	Descripció
I18NService	net.opentrends.openframe.services.i18n	Interfície del servei
I18NException	net.opentrends.openframe.services.i18n.exception	Excepció llençada pel servei
Constants	net.opentrends.openframe.services.i18n	Classe de constants del servei



Cal destacar els següens mètodes del servei:

Components Implementació Spring

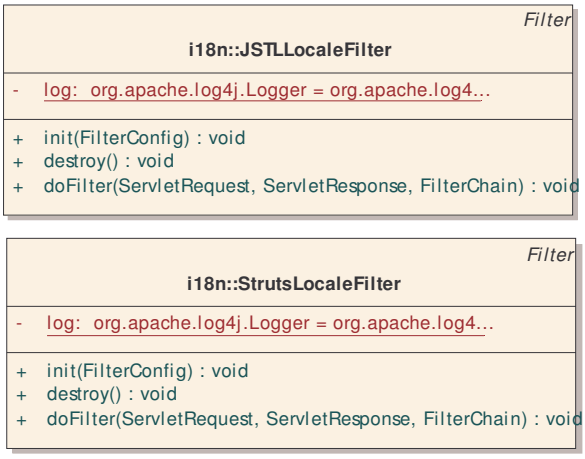
Component	Package	Descripció
SpringI18NServiceImpl	net.opentrends.openframe.services.i18n.impl	Implementació del servei d'internacionalització basada en Spring
CustomDateEditor	net.opentrends.openframe.services.i18n.spring.beans.propertyeditors	Editor específic de dates (veure document' Serveis de Presentació' per més referència)



Components Integració amb la Capa de Presentació

Per a la internacionalització en la capa de presentació s'ofereixen 2 filtres que realitzen la inicialització de les variables de JSTL i Struts per a que poguem fer servir la seva internacionalització.

Component	Package	Descripció
JSTLLocaleFilter	net.opentrends.openframe.services.web.i18n	Filtre web d'integració amb JSTL
StrutsLocaleFilter	net.opentrends.openframe.services.web.i18n	Filtre web d'integració amb Struts



2.2. Instal·lació i Configuració

Instal·lació

La instal·lació del servei requereix de la utilització de la llibreria 'openFrame-services-i18n' i les dependències indicades a l'apartat 'Introducció-Versions i Dependències'.

Configuració

Per a configurar el servei de multi idioma s'han de manipular els següents fitxers:

1) Definició de la font de missatges on el servei cercarà les traduccions
2) Definició del servei (típicament applicationContext.xml)
3) Definició dels fitxers de traduccions (típicament els fitxers dels diferents idiomes: application.properties, application_XX_XX.properties)
4) Definició de la Integració amb el servei de presentació (web.xml)

Definició de la Font de Missatges

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

Mitjançant la definició de la font de missatges podem especificar la llista de fitxers internacionalitzats (resourceBundles) on el servei cercarà les traduccions.

Com a class del bean podem especificar els següents valors:

- **ReloadableResourceBundleMessageSource**

Package: 'org.springframework.context.support'

Està basat en la classe 'Properties' de Java, que permet recarregar els missatges en temps d'execució de l'aplicatiu.

- **ResourceBundleMessageSource**

Package: 'org.springframework.context.support'

Està basada e**n la classe 'ResourceBundle' que no permet la recàrrega dels missatges en temps d'execució. El fet de recarregar els missatges no és crític, ja que depèn de la forma de desplegar l'aplicació en el nostre servidor.**

Per totes les classes indicades podem especificar les següents propietats:

Propietat	Requerit	Descripció
baseNames	Sí	Llista de referències a fitxers. Usar el tag '<list>' i per cada fitxer especificar un tag '<value>' que contingui el path del fitxer (**no s'ha d'incloure l'extensió .properties dels fitxers**). Recomanació: Especificar el valor de cada fitxer mitjançant 'classpath:rutaFitxer' per cercar a partir del classpath Veure 'Definició dels Fitxers de Traducció' per configurar cadascun dels fitxers aquí referenciats.

```
<bean id="messageSource"
class="org.springframework.context.support.ResourceBundleMessageSource">
   <!-- Fitxers de multi idioma -->
   <property name="basenames">
          <list>
                 <value>application</value>
                 <value>fitxer 1</value>
                 ...
                 <value>fitxer "n"</value>
          </list>
   </property>
</bean>
```

Definició del Servei

```
<bean id="i18nService"
      ...>
```

La definició del servei implica indicar quina implementació específica es farà servir

Atributs:

Atributs	Requerit	Descripció
class	Sí	Implementació concreta del servei a utilitzar Opcions: • net.opentrends.openframe.services.i18n.impl.SpringI18nServiceImpl

- **SpringI18nServiceImpl**

  Per aquesta classe podem definir les següents propietats:

Propietat	Requerit	Descripció
messageSource	Sí	Referència a la font de missatges definida al pas previ
defaultLocale	No	Permet especificar quin és el Locale (idioma) per defecte de l'aplicació

Exemple:

```
 <bean id="defaultLocale" class="java.util.Locale">
    <constructor-arg type="String"><value>es</value></constructor-arg>
</bean>
```

```
<bean id="i18nService"
class="net.opentrends.openframe.services.i18n.impl.SpringI18nServiceImpl">
   <!-- Propietat que referència a la font de missatges -->
   <property name="messageSource" ref="messageSource"/>
   <property name="defaultLocale" ref="defaultLocale"/>
</bean>
<!-- Font de missatges -->
<bean id="messageSource"
class="org.springframework.context.support.ResourceBundleMessageSource">

...
</bean>
```

Definició dels Fitxers de Traducció

Ubicació:

Els fitxers s'han de trobar en el classpath de l'aplicació. Es pot configurar la seva localització mitjançant la propietat baseNames (veure 'Definició de la Font de Missatges').

En el cas que especifiquem com a 'baseName' 'i18n/application' i ens trobem en una aplicació Web la cerca del fitxer es realitzarà al directori 'WEB-INF/classes/i18n'.

Explicació:

```
aplicacio.modul.pantalla.key=valor
...
```

Els fitxers de traducció són els fitxers que contenen els missatges en els diferents idiomes. Es basen en l'ús de la internacionalització de Java (i18n o Internationalization).

Aquests fitxers s'han de construir tenint en compte que:

- La seva extensió ha de ser '.properties' (es denominen resource bundles).
- El seu contingut es basa en l'ús de parells clau-valor tal i com es mostra a continuació:

```
#fitxer de multi idioma application.properties
clau_1=valor_1
clau_2=valor_2
...
```

- Per cada idioma que volguem suportar cal crear un nou fitxer. Aquest fitxer tindrà la següent nomenclatura:

  'nomFitxer_*lg_pa*.properties',

  on 'lg correspon al codi del llenguatge i 'pa' correspon al codi del país pel qual volem definir els literals. El codi del país és opcional, pel que podem definir un fitxer en el format:

  Es poden consultar quines són les codificacions per cada llenguatge i país a:

  'http://www.unicode.org/unicode/onlinedat/languages.html'
  'http://www.unicode.org/unicode/onlinedat/countries.html'

- En cas de que l'idioma escollit per l'usuari no correspongui amb cap fitxer, es pot crear un fitxer per defecte. Aquest fitxer tindrà el format 'nomFitxer.properties'.

Exemple:

- 'applicationResources_en.properties', per representar els literals en anglès
- 'applicationResources_es.properties', per representar els literals en castellà
- 'applicationResources_ca.properties', per representar els literals català
- 'applicationResources.properties', per representar els literals de qualsevol altre idioma

Important

És important que es proporcioni un fitxer de recursos per defecte. Aquest serà el fitxer utilitzat pel servei en cas de que el llenguatge escollit per l'usuari no correspongui a cap dels fitxers de recursos definits.

En cas de que per un llenguatge determinat existeixi un fitxer específic, l'obtenció del literal a partir d'una clau s'obtindrà des d'aquest fitxer. Si no es troba la clau en cap cas s'anirà al fitxer per defecte. Així doncs és important que es mantinguin actualitzats els fitxers de recursos per cada idioma de l'aplicació per evitar problemes.

Definició de la Integració amb el servei de presentació

Fitxer de configuració: web.xml

Ubicació proposada: <PROJECT_ROOT>/src/main/webapp/WEB-INF/web.xml
NOTA: Aquesta ubicació és la proposada en cas d'utilització de Maven

- Struts Locale Filter

```
<web-app>
      <filter>
      ...
```

Aquest filtre s'encarrega de integrar el Servei d'Internacionalització per tal que des de Struts es pugui fer ús de fitxers d'internacionalització.

Atributs:

Atribut	Requerit	Valor
filter-name	Sí	Nom del filtre Exemple: Struts Locale Filter
filter-class	Sí	net.opentrends.openframe.services.web.i18n.StrutsLocaleFilter

```
<filter>
   <filter-name>Struts Locale Filter</filter-name>
   <filter-class>
   net.opentrends.openframe.services.web.i18n.StrutsLocaleFilter
   </filter-class>
</filter>
```

- JSTL Locale Filter

  Aquest filtre s'encarrega de integrar el Servei d'Internacionalització per tal que es pugui fer servir des de tags JSTL de tipus fmt

  Atributs:

Atribut	Requerit	Descripció
filter-name	Sí	Nom del filtre Exemple: Struts Locale Filter
filter-class	Sí	Usar 'net.opentrends.openframe.services.web.i18n.JSTLLocaleFilter'

```
<filter>
   <filter-name>JSTL Locale Filter</filter-name>
   <filter-class>
          net.opentrends.openframe.services.web.i18n.JSTLLocaleFilter
   </filter-class>
</filter>
```

2.3. Utilització del Servei

Canviar l'Idioma de l'Usuari

Es pot canviar l'idioma d'un usuari passant un paràmetre a la request amb nom 'set-locale'. El valor d'aquest paràmetre ha de tenir el següent format:

'llenguatge_país',

on llenguatge correspon a la codificació del llenguatge i país a la seva codificació

També es pot passar únicament el llenguatge.

Obtenció de Traduccions

Podem obtenir les traduccions d'un missatge amb els mètodes proporcionats per la interfície:

- ***`public String getMessage(String code)`*: retorna un missatge a partir de la seva clau i del 'Locale' per defecte *(getDefaultLocale())*.**
- ***`public String getMessage(String code, Locale locale)`*: retorna un missatge a partir de la seva clau i un 'Locale'**
- ***`public String getMessage(String code, Object[] args)`*: retorna un missatge a partir de la seva clau, uns arguments per construir un el missatge (típicament els paràmetres {0},{1}, etc) i del 'Locale' per defecte**

- ***public String getMessage(String code, Object[] args, Locale locale)*: retorna un missatge a partir de la seva clau, uns arguments per construir un el missatge (típicament els paràmetres {0},{1}, etc) i un 'Locale'**

En els mètodes en el que no passem el Locale, es farà ús del llenguatge escollit per l'usuari (veure anterior apartat). En cas de no trobar-se cap llenguatge s'usarà el llenguatge per defecte (definit com a propietat del servei 'defaultLocale' o en cas de no trobar-se segons el locale de la màquina virtual).

2.4. Integració amb Altres Serveis

El Servei d'Internacionalització openFrame es tracta d'un servei àmpliament utilitzat des de varis components de la capa de Presentació, principalment en el Servei de Tags.

2.5. Preguntes Freqüènts

3. Exemples

3.1. Exemple de Test Unitari

Com a exemple d'utilització es mostra cóm podem fer una prova unitària:

1. Creem la classe de test 'I18nServiceTest' que extèn de 'TestCase' i afegim un mètode 'testI18N()'

```
public class I18nServiceTest extends TestCase {
    ...
    public void testI18N() {
   ...
    }
}
```

2. Afegim el fitxer de configuració del servei (applicationContext.xml)

```
<beans>
   <bean id="i18nService"
class="net.opentrends.openframe.services.i18n.impl.SpringI18nServiceImpl">
          <property name="messageSource" ref="messageSource"/>
   </bean>
   <bean id="messageSource"
class="org.springframework.context.support.ResourceBundleMessageSource">
          <property name="basenames">
                 <list>
                        <value>application</value>
                 </list>
          </property>
   </bean>
</beans>
```

i els fitxers dels diferents idiomes (application_XX_XX.properties)

```
#application.properties (fitxer per defecte)
var1=Hola!

#application_en_US.properties (fitxer pel 'Locale' 'US')
var1=Attaaaaaack!

#application_en_GB.properties (fitxer pel 'Locale' 'GB')
var1=Hi!
```

3. Implementem el nostre métode de test:

```
    public void testI18N() {
        ClassPathXmlApplicationContext appContext =
                new ClassPathXmlApplicationContext(
                                    "applicationContext.xml");
        BeanFactory factory = (BeanFactory) appContext;

     /**
      * Busquem el servei
      */
        I18nService service =
         (I18nService)factory.getBean("i18nService");

        try {
            assertNotSame("Hola!",service.getMessage("var1"));
            assertNotSame("Hi!",service.getMessage("var1",Locale.UK));

assertNotSame("Attaaaaaack!",service.getMessage("var1",Locale.US));
        }
        catch(I18nServiceException ex){
            assertTrue("var1 not found",false);
        }
    }
```

4. Annexos