Generalitat de Catalunya
Centre de Telecomunicacions
i Tecnologies de la Informació

Framework Corporatiu J2EE

Servei de manipulació de formats OLE

Versió 1.2

Barcelona, 13 / febrer / 2007

Històric de modificacions

Data	Autor	Comentaris	Versió
13/01/2006	Atos Origin, sae openTrends	Versió inicial del document	1.0
13/02/2007	Atos Origin, SAE	Versió 1.2 d'OpenFrame	1.2

Llegenda de Marcadors

Índex

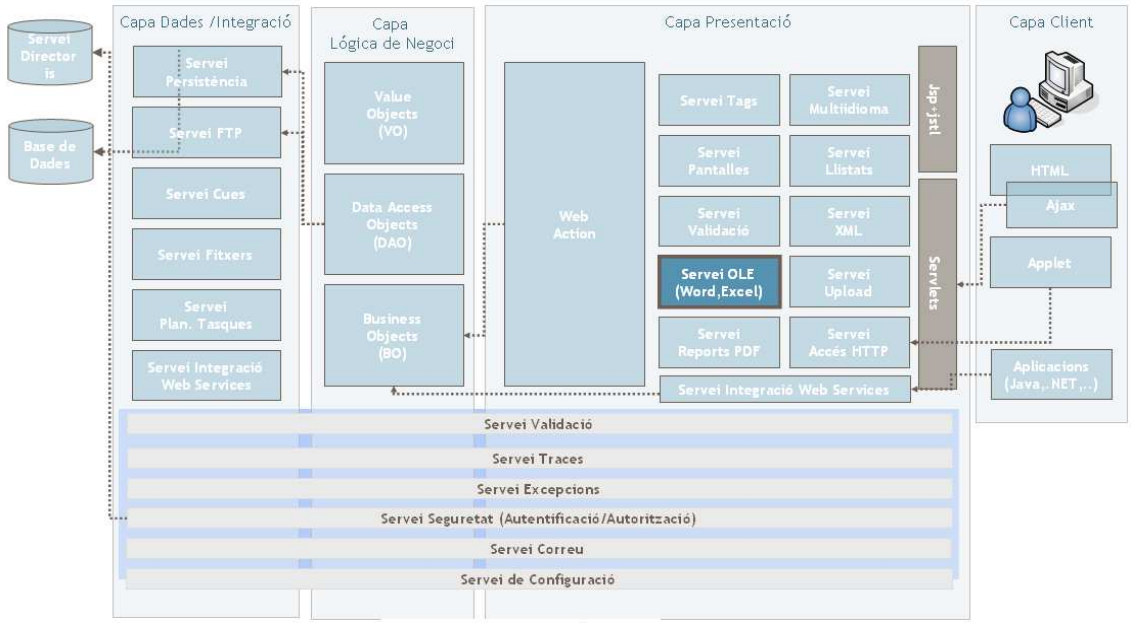
1. Introducció

1.1. Propòsit

El servei de manipulació de formats OLE (Excel i Word) de openFrame permet llegir, crear, modificar i mostrar documents de Microsoft amb formats basats en OLE.

1.2. Context i Escenaris d'Ús

El servei de manipulació de formats OLE es troba dins dels serveis de presentació d'openFrame.



El seu ús és necessari en cas de voler manipular/mostrar documents amb format OLE.

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 addicionals. 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. Versions

No s'han produït canvis respecte la versió 1.0.

1.3.2. Dependències Bàsiques

Nom	Tipus	Versió	Descripció
openFrame-core	jar	1.0	
openFrame-services-logging	jar	1.0	Utilitzar també les dependències del Servei de Logging
openFrame-services-exceptions	jar	1.0	Utilitzar també les dependències del Servei d'Excepcions
spring	jar	1.2.5	http://www.springframework.org
POI	jar	3.0	Llibreria open-source que facilita la manipulació de documents Excel.
POI-scratchpad	jar	3.0	Extensió del POI específic per documents Word.
servlet-api	jar	2.4	

En cas de utilitzar un servei de openFrame és important que es facin servir també les dependències del servei en qüestió.

1.3.3. Dependències Addicionals

- Proves Unitàries del Servei

Nom	Tipus	Versió	Descripció
junit	jar	3.8.1	
Spring-mock	jar	1.2.5	http://www.springframework.org

Veure l'apartat 'Instal·lació i configuració' per a més detall.

1.4. A qui va dirigit

Aquest document va dirigit als següents perfils:

- Programador. Per conèixer l'ús del servei.
- Arquitecte. Per conèixer quins són els components i la configuració del servei.

1.5. Documents i Fonts de Referència

[1] POI http://jakarta.apache.org/poi/

1.6. Glossari

OLE

Object Linking and Embedding, es tracta d'un format de documents propietari de Microsoft utilitzat en els productes Excel i Word des de la versió 97 fins a la 2002.

POI

Poor Obfuscation Implement, es tracta d'un projecte *open source* de Jakarta que pretén facilitar la manipulació de documents OLE des de Java.

2. Descripció Detallada

2.1. Arquitectura i Components

openFrame ofereix una arquitectura d'ús de manipulació de documents OLE totalment deslligada de qualsevol implementació.

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 de les interfícies basada en POI.

> ***Nota***
> Les versions de documents Microsoft que es basen en OLE són des de l'Excel/Word 97 fins a l'Excel/Word 2002.

2.1.1. Interfícies i Components Genèrics

El servei d'utilització de manipulació de documents OLE defineix les següents interfícies i components:

Component	Package	Descripció
OleController	net.opentrends.openframe.services.ole	Controlador de Spring necessari per renderitzar la visualització d'un document OLE com una view. **Serà cridat, pel framework quan sigui necesari.**
ExcelViewIF	net.opentrends.openframe.services.ole	Interfície del servei pels tractament de documents Excel. **Ofereix entre d'altres (veure la documentació disponible al Javadoc per a més referència):** • **protected wrappedBuildExcelDocument(Map model, HSSFWorkbook workbook, HttpServletRequest request, HttpServletResponse response): les subclasses han d'implementar el mètode per renderitzar el document Excel segons les dades del model.** • **protected getCell(HSSFSheet sheet, int row, int col): mètode per obtenir una cel·la segons una fulla, fila i columna.** • **protected setText(HSSFCell cell, String text): mètode per assignar un text en una cel·la.**

Component	Package	Descripció
WordViewIF	net.opentrends.openframe.services.ole	Interfície del servei pels tractament de documents Word. **Ofereix entre d'altres (veure la documentació disponible al Javadoc per a més referència):** • **public getText(WordDocument wordDocument): retorna un String amb el texte d'un document Word.** • **Protected insertAfter(WordDocument wordDocument, String text): mètode per inserir texte al final d'un document Word.** • **Protected insertBefore(WordDocument wordDocument, String text): mètode per inserir texte al principi d'un document Word.**

2.1.2. Components implementació per POI

Component	Package	Descripció
WrapperWordView	net.opentrends.openframe.services.ole.impl	Implementació de la interfície WordViewIF pel component POI.
WrapperExcelView	net.opentrends.openframe.services.ole.impl	Implementació de la interfície ExcelViewIF pel component POI.
WordDocument	net.opentrends.openframe.services.ole.impl	Classe necesària per la implementació de la interfície WordViewIF pel component POI.

Veure la documentació disponible al Javadoc per a més referència.

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-ole' i les dependències indicades a l'apartat 'Introducció - Versions i Dependències'.

2.2.2. Configuració

Per a configurar el servei de manipulació de formats OLE s'han d'actualitzar els següents fitxers:

La configuració implica els següents pasos:

1) Definició bàsica del servei. Cal modificar els següents fitxers
   - web.xml
   - openFrame-services.xml
   - struts-config.xml
   - application-servlet.xml

2) Definir els constructors de documents OLE. Cal modificar el fitxer 'views.xml'

Definició Bàsica del Servei

- Definició del mapeig de peticions url de tipus '.doc' i '.xls'

  Fitxer de configuració: web.xml

  Ubicació: <PROJECT_ROOT>/src/main/webapp/WEB-INF

  En aquest punt es defineix que tota petició realitzada amb extensió '.doc' o '.xls' les resoldrà el servlet principal.

  Definir el següent codi:

```
...
<servlet>
        <servlet-name>application</servlet-name>
        <servlet-class>
                org.springframework.web.servlet.DispatcherServlet
        </servlet-class>
        <load-on-startup>2</load-on-startup>
  </servlet>
...
  <servlet-mapping>
        <servlet-name>application</servlet-name>
```

```
    </servlet-mapping>
    <servlet-mapping>
            <servlet-name>application</servlet-name>
            <url-pattern>*.xls</url-pattern>
    </servlet-mapping>
...
```

- Definició del resolver que defineix el fitxer de definició de reports

  Fitxer de configuració: openFrame-services-ole.xml

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

  Definir el següent codi:

```
...
<!-- OLE service  -->
<bean id="oleResolver"
class="org.springframework.web.servlet.view.XmlViewResolver" >
        <property name="location" value="classpath:spring/openFrame-services-
views.xml" />
</bean>
...
```

- Definició dels forwards de Struts per usar el servei des d'una Action

  Fitxer de configuració: struts-config.xml

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

  Definir el següent codi:

```
...
<global-forwards>
   ...

   <!-- Forward a un objecte OLE -->
   <forward name="oleXLS" path="/ole.xls" redirect="true" />
   <forward name="oleDOC" path="/ole.doc" redirect="true" />

   ...
</global-forwards>
...
```

- Definició del resolver que defineix el fitxer de definició de reports

Fitxer de configuració: openFrame-services-ole.xml

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

Definir el següent codi:

- Definició del control·lador de tractament dels reports

Fitxer de configuració: action-servlet.xml

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

Definir el següent codi:

```
...
<bean id="urlMapping"
class="org.springframework.web.servlet.handler.SimpleUrlHandlerMapping">
   ...
   <property name="mappings">
          <props>
                 ...
                 <prop key="ole.xls">oleController</prop>
                 <prop key="ole.doc">oleController</prop>
          </props>
   </property>
</bean>
...

<bean id="oleController"
   class="net.opentrends.openframe.services.ole.OleController" />
...
```

Definició dels constructors dels documents OLE

Fitxer de configuració: openFrame-services-views.xml

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

En aquest fitxer es defineixen els beans que renderitzen els documents amb format OLE. Cadascuna de les classes de document que extenguin de 'WrapperWordView' o 'WrapperExcelView' (veure apartat 'Utilització del Servei') seran definides en aquest apartat.

Atributs:

Atribut	Requerit	Descripció
class	Sí	Es tracta de la classe que renderitza de les dades a format OLE. Aquesta classe la definir-se s'haurà extés de WrapperWordView o

Atribut	Requerit	Descripció
		WrapperExcelView.

Per cada bean es poden configurar les següents propietats:

Propietat	Requerit	Descripció
beanName	No	Nom del tipus de classe contingut a la llista del 'beanList'
beanList	No	Nom de la llista que conté la informació. Aquesta informació serà introduida des de l'Action
url	Sí	Localització del fitxer que farà de plantilla. En el cas d'Excels serà opcional. NOTA: en el cas dels Words s'ha d'utilitzar SEMPRE un document ja creat (encara que sigui en blanc).

Tant 'beanName' com 'beanList' són informacions d'ajuda a la vista per accedir a les dades que li prepari el Action. En qualsevol cas, es poden passar paràmetres entre el Action i la vista de qualsevol tipus sense definir aquestes propietats.

Exemple:

```
<bean id="itemsWordView"
class="net.opentrends.openframe.samples.jpetstore.ole.views.ItemsWordView">
   <property name="beanName"
value="net.opentrends.openframe.samples.jpetstore.model.Item"/>
   <property name="beanList" value="itemList"/>
   <property name="url" value="classpath:ole/template.doc"/>
   <property name="logService" ref="loggingService"/>
</bean>
```

2.3. Utilització del Servei

2.3.1. Generació d'un document amb format OLE

A continuació es mostren els pasos necessaris per generar un document en format OLE:

1) Crear el Action si no existeix i la funció que generarà la informació necessària per la vista.

```
public ActionForward viewOleWord(ActionMapping mapping, ActionForm form,
            javax.servlet.http.HttpServletRequest request,
```

```
            ...
            List list = ((HibernateDAO)this.dao).findAll();
            if(list!=null){
                Map model = new HashMap();
                Iterator it = list.iterator();
                while(it.hasNext()){
                    Item i = (Item)it.next();
                    Hibernate.initialize(i.getProductid().getCategory());
                }

                model.put("itemList",list);❶

    request.getSession().setAttribute(OleController.OLE_CONTROLLER_MODEL,model);❷

    request.getSession().setAttribute(OleController.OLE_CONTROLLER_VIEWID,"itemsWordVi
    ew");❸
            }
            return mapping.findForward("oleController")❹;
        }
```

Una vegada obtinguda la informació a mostrar al document, haurem de realitzar els següents pasos:

- ❶ Introduir en un mapa (Map) la informació que voldrem mostrar. A l'exemple a dalt mostrat 'itemList' contindrà la llista dels items
- ❷ Ficar a l'atribut de sessió 'OleController.OLE_CONTROLLER_MODEL' el mapa creat amb la informació
- ❸ Introduir a l'atribut de sessió 'OleController.OLE_CONTROLLER_VIEWID' el nom de la vista que farà servir aquesta informació. Aquest nom correspon al definit al fitxer de configuració 'openFrame-services-views.xml'
- ❹ Finalment fer un forward a 'oleController'. Aquest és el control·lador que usarà la vista. Recordar que aquest s'ha definit al fitxer de configuració:

```
<bean id="urlMapping"
class="org.springframework.web.servlet.handler.SimpleUrlHandlerMa
pping">
   ...
   <property name="mappings">
          <props>
                 ...
                 <prop key="ole.xls">oleController</prop>
                 <prop key="ole.doc">oleController</prop>
          </props>
   </property>
</bean>
...

<bean id="oleController"
class="net.opentrends.openframe.services.ole.OleController" />
```

2) Crear la vista. Aquesta ha d'heretar de 'net.opentrends.openframe.services.ole.impl.WrapperWordView' si es vol un document Word o de 'net.opentrends.openframe.services.ole.impl .WrapperExcelView' si volem generar un document Excel

   La vista ha d'implementar el mètode 'wrappedBuildWordDocument' o 'wrappedBuildExcelDocument' segons es tracti d'una vista de Word o de Excel.

   En el paràmetre 'model' d'aquests mètodes, la vista pot accedir a les dades preparades per l'Action. La vista usarà els mètodes proporcionats per l'API de POI per generar el contingut.

```
 protected void wrappedBuildWordDocument(
            Map model, WordDocument wordDocument, HttpServletRequest request,
HttpServletResponse response)
            throws OleServiceException{

        List list = (List)model.get(this.getBeanList());

        if (logService!=null)
          this.logService.getLog(this.getClass()).debug("Found
list?"+(list!=null));



        if(list!=null){
            Iterator it = list.iterator();
            Range range = new Range(0, wordDocument.characterLength(),
wordDocument);
            while(it.hasNext()){
                Item item = (Item)it.next();
                CharacterProperties props = new CharacterProperties();
                props.setBold(true);
                range = range.insertAfter(" Item ID="+item.getId()+"\r", props);
                props.setBold(false);
                range = range.insertAfter(" Name: ",props);
                props.setItalic(true);
                range = range.insertAfter(" "+item.getAttr1());
                props.setItalic(false);
                range = range.insertAfter(", Product: ",props);
                props.setItalic(true);
                range = range.insertAfter("
"+item.getProductid().getName(),props);
                props.setItalic(false);
                range = range.insertAfter(", Category: ",props);
                props.setItalic(true);
                range = range.insertAfter("
"+item.getProductid().getCategory().getName()+"\r",props);
            }
        }
    }
```

3) Definir la vista al fitxer de configuració (veure apartat 'Configuració')

Per últim, cal recordar que la invocació funciona segons el mecanisme general d'indicació del paràmetre 'reqCode'.

2.4. Eines de Suport

2.5. Integració amb Altres Serveis

2.6. Preguntes Freqüents

3. Exemples

3.1. Generar un document Excel en format OLE

A continuació, com exemple, es mostren les passes a seguir per generar un document Excel en format OLE.

3.1.1. Definir el document a modelar en el servei (views.xml)

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN"
"http://www.springframework.org/dtd/spring-beans.dtd">

<beans>

   ...
   <bean id="WidgetExcelExample"
class="net.opentrends.samples.ole.WidgetListExcelView" >
          <property name="beanName" value="WidgetExcelExampleBean" />
          <property name="beanList" value="WidgetExcelExampleList" />
   </bean>
   ...

</beans>
```

3.1.2. Modelar les dades a mostrar

En l'*Action* es modelaran les dades que, posteriorment, es renderitzaran donant com a resultat el document en format OLE.

```
/**
 * Using OLE (Excel)
 * @param request
 * @param response
 * @return ActionForward
 */
public ActionForward viewOleExcel(Category vo, StrutsContext context) {
   // Get data for render Excel document
   Map model = getWidgetModel();

   // Save data in session
   context.getRequest().getSession()
    .setAttribute(OleController.OLE_CONTROLLER_MODEL, model);

   // Set view name to render
   context.getRequest().getSession()
    .setAttribute(OleController.OLE_CONTROLLER_VIEWID,
                  "WidgetExcelExample");

   // Forward to OLE Controller
   return context.getActionMapping().findForward("oleXLS");
}
```

```
private Map getWidgetModel() {
   Map model = new HashMap();
   Collection beanData = getWidgetData();
   model.put("WidgetExcelExampleList", beanData);
   return model;
}

protected List getWidgetData() {
   List list = new ArrayList();
   for (int x = 0; x < 10; x++) {
          Widget bean = new Widget();
          bean.setId(x);
          bean.setName("openFrame");
          bean.setSize(10*x);
          list.add(bean);
   }
   return list;
}
```

3.1.3. Renderitzar les dades a mostrar

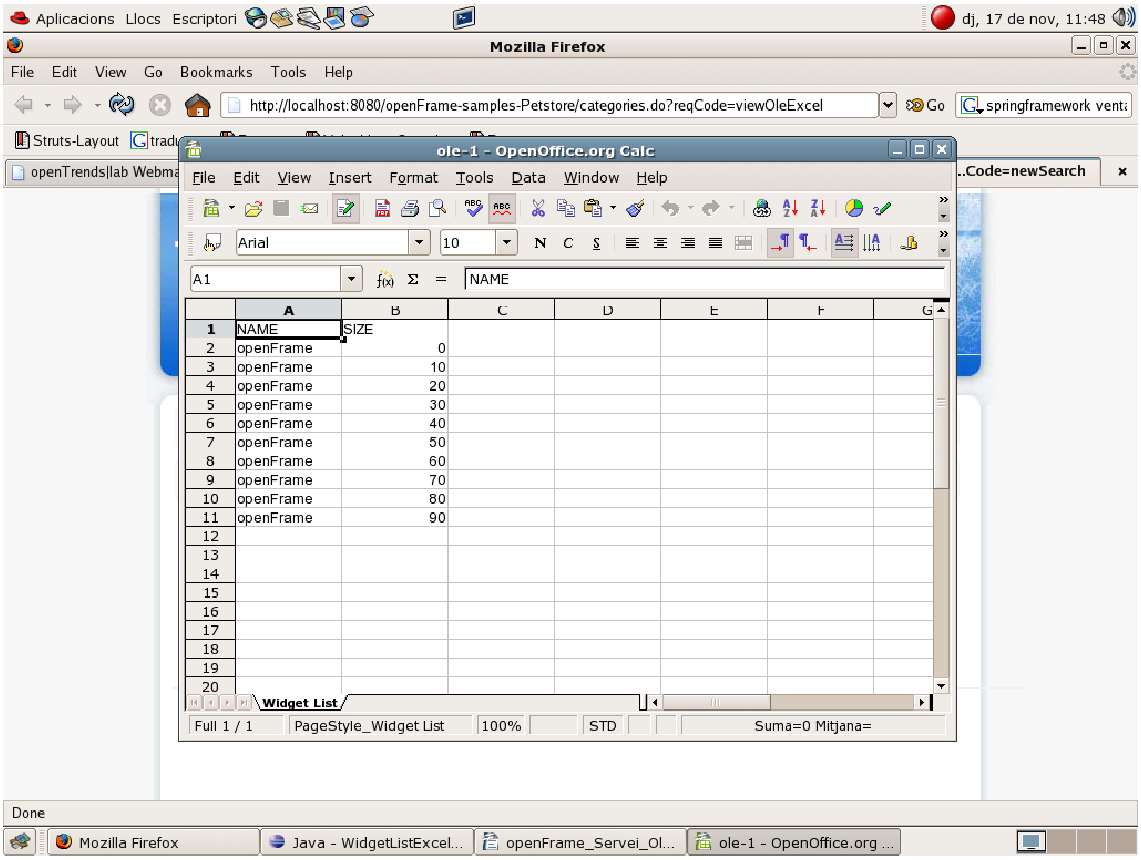
S'ha de definir una classe que renderitzi les dades a mostrar en format OLE. Aquesta classe ha d'extendre de *WrapperExcelView* i utilitza directament l'API de POI (HSSF per Excel i HWPF per Word) per renderitzar el document.

```
package net.opentrends.samples.ole;

/** imports */


/**
 * View to generate an Excel document of a list of Widgets
 */
public class WidgetListExcelView extends WrapperExcelView {
   protected static final short WIDGET_NAME_COLUMN = 0;
   protected static final short WIDGET_SIZE_COLUMN = 1;

   protected void wrappedBuildExcelDocument(Map model,
HSSFWorkbook workbook,HttpServletRequest request, HttpServletResponse response) {
          // CREATE THE SHEET
          HSSFSheet sheet = workbook.createSheet("Widget List");
          sheet.setDefaultColumnWidth((short) 12);

          // GETCELL: getCell(SHEET, ROW, COLUMN);
          short currentRow = 0;

          // WRITE ROW FOR HEADER
          HSSFCell header0 = getCell(   sheet,
                                        currentRow,
                                        WIDGET_NAME_COLUMN);
          setText(header0, "NAME");

          HSSFCell header1 = getCell(   sheet,
                                        currentRow,
                                        WIDGET_SIZE_COLUMN);
          setText(header1, "SIZE");

          // Get data from model with key "beanList"
          List widgetList = (List) model.get(getBeanList());
          Iterator widgetListIterator = widgetList.iterator();

          while (widgetListIterator.hasNext()) {
```

```
                    currentRow++;
                    Widget widget = (Widget) widgetListIterator.next();
                    HSSFRow row = sheet.createRow(currentRow);

                    row.createCell(WIDGET_NAME_COLUMN)
                     .setCellValue(widget.getName());

                    row.createCell(WIDGET_SIZE_COLUMN)
                     .setCellValue(widget.getSize());
            }

    }
}
```

3.1.4. Invocació

Per invocar la generació del fitxer Excel es seguirà el procés general de passar com a reqCode el nom del mètode de l'Action que volem executar, en aquest cas 'viewOleExcel'.



4. Annexos