Creació Client CXF per l'STD

A qui va dirigit

Aquest Howto va dirigit a tots aquells perfils tècnics encarregats de la implementació de l'accés als serveis del WS del STD (Sistema de Transformació de Documents).

Versió de Canigó

Aquest Howto està dissenyat per que aplicacions que no són Canigó puguin utilitzar un client java per l'STD.

Introducció

La construcció d'un client java WS és un procés relativament senzill, tot i així sempre existeixen poden aparèixer problemes típics, com per exemple, els relacionats amb la tecnologia utilitzada (CXF, Axis, XmlBeans, etc) o bé per la carència d'exemples.

A tal efecte es presenta aquest HowTo on s'introdueix una forma per crear un client CXF per consumir els diferents serveis del STD així com una llista d'exemples de com utilitzar-lo.

Com ja s'ha comentat, en la creació de clients per WebServices existeixen altres tecnologies diferents a CXF, com podrien ser Xmlbeans o Axis. Els motius per escollir CXF han sigut la senzillesa del procés i la baixa complexitat del client generat. Això **no significa que es puguin generar clients per l'STD amb altres tecnologies**, però des del CS Canigó es recomana fer-ho amb CXF sempre que sigui possible.

En cas de necessitat d'utilització d'un altre tecnologia es poden consultar els problemes derivats a través de la bústia del portal de Frameworks i Solucions d'Arquitectura (oficina-tecnica.canigo.ctti@gencat.cat) o bé a través de l'eina Jira del CTTI en http://cstd.ctti.gencat.cat/jiracstd.

Construcció del Client

Introducció

L'objectiu del Howto és poder crear un Jar, una llibreria, amb el contingut del client. D'aquesta forma és més fàcil integrar-lo en qualsevol altre aplicació com a dependència.

Per poder crear el client en CXF aquest Howto es basarà en l'ús d'un plugin de Maven2 anomenat "cxf-codegen-plugin". Per aquest motiu, el projecte que contindrà el client serà de tipus Maven2.

Consideracions Inicials: El procediment que és descriurà preveu que l'entorn de treball sigui l'entorn de treball de Canigó; Eclipse 3.6, JDK 1.6 i Maven 2.2.1.
Es pot descarregar l'entorn de la següent url:
http://canigo.ctti.gencat.cat/confluence/display/CAN/Baixar+binaris en la secció "Creació entorn local".

Creació Projecte

1.- Dintre de l'eclipse anar a "File" -> "New" -> "Project".
2.- Seleccionar el "Wizard" de Maven -> "Maven Project" i prémer "Next".

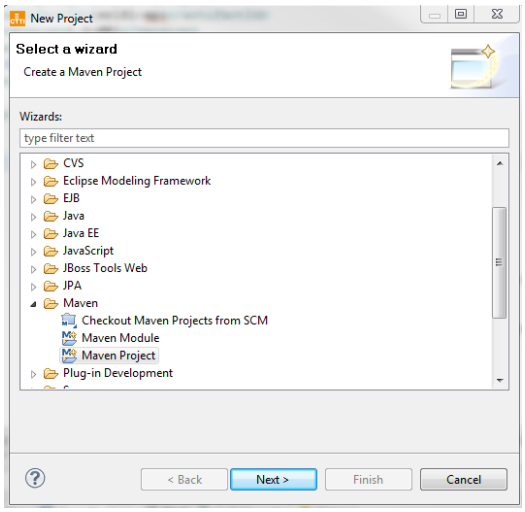

Figura 1. Wizard projecte Maven2.

3.- Es selecciona la ubicació del projecte i es prem "Next".

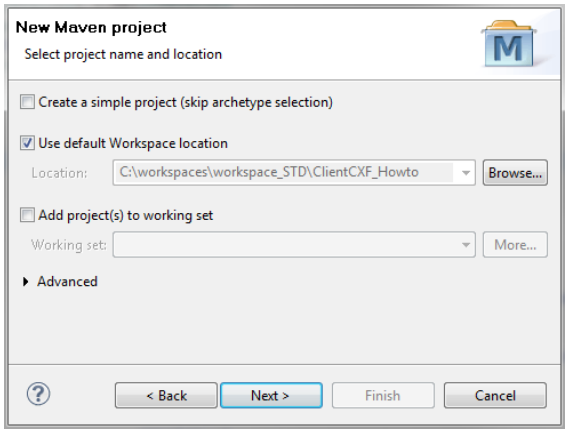

Figura 2. Ubicació projecte

4.- Seleccionar l'arquetip del projecte. Per defecte es queda seleccionat "maven-arquetype-quickstart" que és el que hem de deixar (NOTA: la llista d'arquetips pot trigar a carregar uns segons). Prémer "Next".

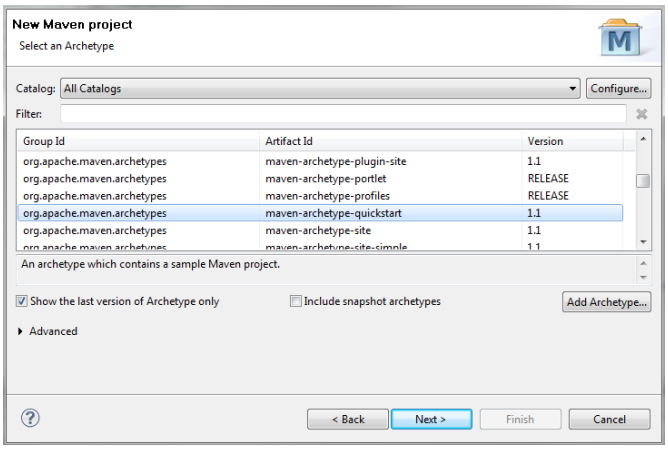

Figura 3. Selecció d'arquetips

5.- Es defineixen les coordenades maven de la llibreria, grupId, artifactId i versió. Es prem "Finish".

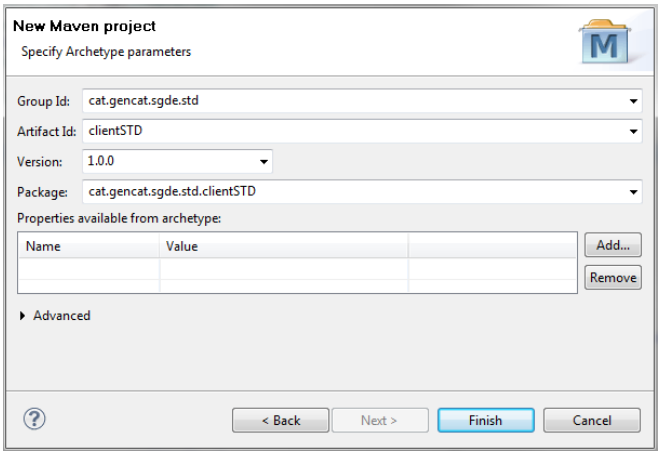

Figura 4. Creació de coordenades Maven del client.

6.- Aquest procés crea el projecte en el workspace de l'eclipse que s'ha seleccionat. Una vegada creat s'hauran de fer les següents accions;

6.1.- Assegurar la instal·lació de maven. Anar a l'opció "Window" -> "Preferences" del l'eclipse. Seleccionar l'opció "Maven";

- Assegurar-se que el check "Offline" es troba marcat.
- En la sub-opció "Installations" seleccionar la instal·lació de Maven2.2.1. (Per defecte és possible que només estigui la de Maven3, amb aquesta versió no es garanteix el correcte funcionament). Si no hi és afegir-la.

6.2.- habilitar el gestor de dependències de maven. Prémer botó dret sobre el projecte creat i clicar en l'opció "Maven".

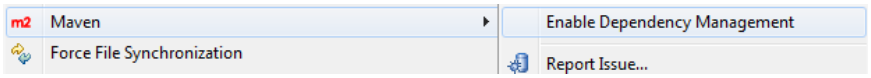

Figura 5. Activació del gestor de dependències de Maven.

Prémer l'opció "Enable Dependency Management".

Modificació del pom.xml

S'han d'afegir les següents dependències;

```
<dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>4.5</version>
        <scope>test</scope>
</dependency>
<dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-test</artifactId>
        <version>3.0.5.RELEASE</version>
        <scope>test</scope>
```

```
</dependency>
<dependency>
        <groupId>org.apache.cxf</groupId>
        <artifactId>cxf-rt-frontend-jaxws</artifactId>
        <version>${cxf.version}</version>
</dependency>
<dependency>
        <groupId>org.apache.cxf</groupId>
        <artifactId>cxf-rt-transports-http</artifactId>
        <version>${cxf.version}</version>
</dependency>
<dependency>
         <groupId>javax.xml.ws</groupId>
         <artifactId>jaxws-api</artifactId>
         <version>2.1</version>
</dependency>
<dependency>
        <groupId>javax.jws</groupId>
        <artifactId>jsr181-api</artifactId>
        <version>1.0-MR1</version>
</dependency>
```

i els següents plugins;

```
<plugin>
        <groupId>org.apache.cxf</groupId>
        <artifactId>cxf-codegen-plugin</artifactId>
        <version>2.3.2</version>
        <executions>
                <execution>
                        <id>generate-sources</id>
                        <phase>generate-sources</phase>
                        <configuration>
                                <sourceRoot>${basedir}/src/main/java/</sourceRoot>
                                <wsdlOptions>
                                        <wsdlOption>

        <wsdl>http://sgde.intranet.gencat.cat/ServeisInvocacioSTD/services/ServeisSTD?wsd
l</wsdl>
                                        </wsdlOption>
                                </wsdlOptions>
                        </configuration>
                        <goals>
                                <goal>wsdl2java</goal>
                        </goals>
                </execution>
        </executions>
</plugin>
<plugin>
       <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-compiler-plugin</artifactId>
        <configuration>
                <target>1.5</target>
                <source>1.5</source>
        </configuration>
</plugin>
```

D'aquests últims destaca el paràmetre wsdl del plugin "cxf-codegen-plugin". En aquest paràmetre s'ha d'indicar el wsdl del WS objectiu, en l'exemple, el wsdl de l'entorn de PRO del STD.
NOTA: La versió recomanada de CXF és la 2.5.1 (cxf.version)

Primera execució

Per poder compilar el projecte s'han de seguir els següents passos;

Creació Client CXF per l'STD

1.- Prémer botó dret sobre el projecte, seleccionar la opció "Run As" i seleccionar la opció "clean".
2.- En el directori src/main/java apareixeran totes les classes autogenerades del client. Degut al procediment que fa CXF de forma interna, hi ha una classe que no es genera correctament. Aquesta és la "ConfigServiceSTD.java" que li falta un setter d'una de les seves variables.

Per corregir-ho afegir manualment el següent setter a la classe "ConfigServiceSTD" del package "cat.gencat.ctti.canigo.eforms.services.objects.xsd"

```
public void setParametresComposicio(List<Parametre> param) {
        if (parametresComposicio == null) {
                parametresComposicio = new ArrayList<Parametre>(param);
        } else {
                parametresComposicio = param;
        }
}
```

3.- Igual que en el pas 1 però aquesta vegada prémer "Package". Si la compilació va bé el producte final serà un jar amb les classes del client en el seu interior que es podrà trobar dintre del directori "Target".

NOTA: Si per algun motiu s'ha de tornar a compilar el projecte, per l'addicció dels test per exemple, es recomana comentar el plugin de cxf en el pom.xml. Si no es fes això cada execució del tipus "clean package" esborraria i generaria de 0 el client, sense el setter que s'ha afegit de forma manual.

Configuració de l'aplicació pels tests (Opcional)

Aquest punt no és necessari per la creació del client, però si ho és per fer funcionar els JUnits on es posaran les crides al WS d'exemple.

Dintre del directori src/test/resources (crear si no existeix), crear l'estructura [nom_del_package_base del projecte] (ex: cat.gencat.sgde.eformularis.ClientCXF_Howto) i en el seu interior l'arxiu "aplicationContext.xml" amb el contingut;

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns:jaxws="http://cxf.apache.org/jaxws"
        xsi:schemaLocation="http://www.springframework.org/schema/beans
                http://www.springframework.org/schema/beans/spring-beans.xsd
                http://cxf.apache.org/jaxws
                http://cxf.apache.org/schemas/jaxws.xsd">

    <jaxws:client
        id="serveisSTD"
        serviceClass="cat.gencat.ctti.canigo.eforms.std.ws.impl.ServeisSTDPortType"

        address="http://sgde.intranet.gencat.cat/ServeisInvocacioSTD/services/ServeisSTD?
wsdl"
    />
</beans>
```

On com es pot observar, es torna a definir en el camp address el valor del wsdl del WS.

JUnits(Opcional)

Amb l'objectiu de poder provar el client es poden definir JUnits en el projecte que acompliran un doble objectiu:

Creació Client CXF per l'STD

- Primer verificar que el client funciona i té accés al webService des de l'entorn on s'executi.
- Segon, mostrar exemples de consum dels diferents serveis que ofereix l'STD.

Els JUnits aniran en el directori src/test/java (crear si no existeix) i dintre del package [nom_del_package_base del projecte] (ex: cat.gencat.sgde.eformularis.ClientCXF_Howto). Es definiran dintre d'una classe de la següent forma;

```
package cat.gencat.sgde.eformularis.ClientCXF_Howto;

import java.io.ByteArrayOutputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.InputStream;
import java.util.ArrayList;
import java.util.List;

import javax.xml.bind.JAXBElement;
import javax.xml.datatype.DatatypeConfigurationException;
import javax.xml.datatype.DatatypeFactory;
import javax.xml.namespace.QName;

import org.junit.Test;
import org.junit.runner.RunWith;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.test.context.ContextConfiguration;
import org.springframework.test.context.junit4.SpringJUnit4ClassRunner;

import cat.gencat.ctti.canigo.eforms.services.objects.xsd.ConfigCall;
import cat.gencat.ctti.canigo.eforms.services.objects.xsd.ConfigServiceSTD;
import cat.gencat.ctti.canigo.eforms.services.objects.xsd.Parametre;
import cat.gencat.ctti.canigo.eforms.services.objects.xsd.ResultSTD;
import cat.gencat.ctti.canigo.eforms.std.composicio.estampat.segells.xsd.StringSegell;
import cat.gencat.ctti.canigo.eforms.std.ws.impl.ServeisSTDPortType;

/*
 *  ServeisSTDTest Junit test case
 */
@RunWith(SpringJUnit4ClassRunner.class)
@ContextConfiguration(locations={"classpath:cat/gencat/sgde/std/ClientCXF_Howto/aplicationContext.xml
"})
public class ServeisSTDTest extends junit.framework.TestCase {

        private String ambitProves = "[ambit]";
        private String aplicacioProves = "[aplicacio]";

        private final static QName _ParametreClau_QNAME = new
QName("http://objects.services.eforms.canigo.ctti.gencat.cat/xsd", "clau");
    private final static QName _ParametreValorString_QNAME = new
QName("http://objects.services.eforms.canigo.ctti.gencat.cat/xsd", "valorString");
    private final static QName _StringSegellText_QNAME = new
QName("http://segells.estampat.composicio.std.eforms.canigo.ctti.gencat.cat/xsd", "text");

        @Autowired
        ServeisSTDPortType service;

        @Test
        public void TestGenerarCSV() {

                try {
                        System.out.println("[TestGenerarCSV] Inici");
                        //Constructor d'objectes bàsics
                        cat.gencat.ctti.canigo.eforms.services.objects.xsd.ObjectFactory
objectFactory =
                                        new
cat.gencat.ctti.canigo.eforms.services.objects.xsd.ObjectFactory();

                        ConfigCall config = new ConfigCall();

                        JAXBElement<String> ambit =
objectFactory.createConfigCallAmbit(ambitProves);
                        JAXBElement<String> aplicacio =
objectFactory.createConfigCallAmbit(aplicacioProves);

                        config.setAmbit(ambit);
```

Creació Client CXF per l'STD

```
                        config.setAplicacio(aplicacio);

                        File fitxerEntrada = new
File("C:\\tmp\\STD\\ctti\\cscanigo\\documents\\composicio\\in\\prueba_composar.pdf");
                        InputStream inStream = new FileInputStream(fitxerEntrada);

                        ByteArrayOutputStream buffer = new ByteArrayOutputStream();
                        int nRead;
                        byte[] data = new byte[1024];
                        while ((nRead = inStream.read(data, 0, data.length)) != -1) {
                          buffer.write(data, 0, nRead);
                        }
                        buffer.flush();

                        //Invocació del servei
                        ResultSTD result = services.generarCSV(config, buffer.toByteArray(),
"clau");

                        System.out.println("[TestGenerarCSV] Resultat: " +
result.getKey().getValue());
                        System.out.println("[TestGenerarCSV] Estat: " +
result.getStatus().intValue());
                        System.out.println("[TestGenerarCSV] MissatgeError: " +
result.getMissatgeError().getValue());
                        System.out.println("[TestGenerarCSV] TimeStamp: " +
result.getTimeStamp().getValue());
                        System.out.println("[TestGenerarCSV] Final");

                } catch (Exception e) {
                        System.out.println("ERROR en el Test del Servei de generació de CSV");
                        e.printStackTrace();
                }
        }
}
```

A continuació es mostren altres JUnits d'exemple que exploren diferents operacions amb el client sobre el WS del STD.

```
@Test
public void TestComposarPDFSimple() {

        try {

                System.out.println("[TestComposarPDFSimple] Inici");
                //Constructor d'objectes bàsics
                cat.gencat.ctti.canigo.eforms.services.objects.xsd.ObjectFactory objectFactory =
                        new cat.gencat.ctti.canigo.eforms.services.objects.xsd.ObjectFactory();

                ConfigCall config = new ConfigCall();

                JAXBElement<String> ambit = objectFactory.createConfigCallAmbit(ambitProves);
                JAXBElement<String> aplicacio =
objectFactory.createConfigCallAmbit(aplicacioProves);

                config.setAmbit(ambit);
                config.setAplicacio(aplicacio);

                File fitxerEntrada = new
File("C:\\tmp\\STD\\ctti\\cscanigo\\documents\\composicio\\in\\prueba_composar.pdf");
                InputStream inStream = new FileInputStream(fitxerEntrada);

                ByteArrayOutputStream buffer = new ByteArrayOutputStream();
                int nRead;
                byte[] data = new byte[1024];
                while ((nRead = inStream.read(data, 0, data.length)) != -1) {
                  buffer.write(data, 0, nRead);
                }
                buffer.flush();

                ConfigServiceSTD configSTD = new ConfigServiceSTD();

                JAXBElement<byte[]> arxiu =
objectFactory.createConfigServiceSTDInputStreamEntrada(buffer.toByteArray());
                configSTD.setInputStreamEntrada(arxiu);
```

Creació Client CXF per l'STD

```
                JAXBElement<String> plantilla =
objectFactory.createConfigServiceSTDPlantilla("plantilla_marge26_tot");
                configSTD.setPlantilla(plantilla);

                JAXBElement<Float> escalat =
objectFactory.createConfigServiceSTDPorcentatgeEscalat(new Float(1));
                configSTD.setPorcentatgeEscalat(escalat);

                JAXBElement<Float> escaX = objectFactory.createConfigServiceSTDPosicioEscalatX(new
Float(5));
                configSTD.setPosicioEscalatX(escaX);

                JAXBElement<Float> escaY = objectFactory.createConfigServiceSTDPosicioEscalatY(new
Float(5));
                configSTD.setPosicioEscalatY(escaY);

                //carrega de paràmetres de composició
                //1.- Creació llista de paràmetres
                List<Parametre> params = new ArrayList<Parametre>();
                //2.- Creació d'un paràmetre
                Parametre param1 = new Parametre();

                //2.1.- Creació de la clau del primer paràmetre
                JAXBElement<String> clau = new JAXBElement<String>(_ParametreClau_QNAME,
String.class, "text_horiz");
                param1.setClau(clau);

                //2.2.- Creació del valor de la clau (un stringSegell)
                StringSegell stringSegell = new StringSegell();

                //2.2.1.- carrega de les dades del StringSegell
                JAXBElement<String> text = new JAXBElement<String>(_StringSegellText_QNAME,
String.class, "Prova test Horitzontal");
                stringSegell.setText(text);

                //2.2.2.- Carrega del valor de la clau
                JAXBElement<StringSegell> valor = new
JAXBElement<StringSegell>(_ParametreValorString_QNAME, StringSegell.class, stringSegell);
                param1.setValorString(valor);

                //3.- carrega del paràmetre
                params.add(param1);

                //4.- Càrrega de la llista de paràmetres
                configSTD.setParametresComposicio(params);

                configSTD.setGenerarCSV(true);

                //Crida al servei
                ResultSTD result = services.composarPDFSimple(config, configSTD);

                JAXBElement<byte[]> out = result.getArxiu();

                if (out.getValue() != null) {
                        FileOutputStream fos =
                                new FileOutputStream(new
File("C:\\tmp\\STD\\ctti\\cscanigo\\documents\\composicio\\out\\prueba_composar_out.pdf"));
                        fos.write(out.getValue());
                        fos.flush();
                        fos.close();
                }
                System.out.println("[TestComposarPDFSimple] Resultat: " +
result.getKey().getValue());
                System.out.println("[TestComposarPDFSimple] MissatgeError: " +
result.getMissatgeError().getValue());
                System.out.println("[TestComposarPDFSimple] TimeStamp: " +
result.getTimeStamp().getValue());
                System.out.println("[TestComposarPDFSimple] Estat: " +
result.getStatus().intValue());
                System.out.println("[TestComposarPDFSimple] Final");

        } catch (Exception e) {
                System.out.println("ERROR en el Test del Servei de composició de PDF (bytes)");
                e.printStackTrace();
        }
}
```

```
@Test
```

Creació Client CXF per l'STD

```
public void TestComposarPDFRemot() {

        try {

                System.out.println("[TestComposarPDFRemot] Inici");
                //Constructor de objectes bàsics
                cat.gencat.ctti.canigo.eforms.services.objects.xsd.ObjectFactory objectFactory =
                        new cat.gencat.ctti.canigo.eforms.services.objects.xsd.ObjectFactory();
                        String nomFitxerEntrada = "prueba_composar";
                String nomPlantilla = "plantilla_marge26_tot";
                String nomFitxerSortida="prueba_composar_remot_out.pdf";
                Float porcentatgeEscalat = new Float(1);
                Float posicioEscalatX = new Float(5);
                Float posicioEscalatY = new Float(5);
                boolean generarCSV = false;

                ConfigCall config = new ConfigCall();

                JAXBElement<String> ambit = objectFactory.createConfigCallAmbit(ambitProves);
                JAXBElement<String> aplicacio =
objectFactory.createConfigCallAmbit(aplicacioProves);

                config.setAmbit(ambit);
                config.setAplicacio(aplicacio);

                //carrega de paràmetres de composició
                //1.- Creació llista de paràmetres
                List<Parametre> params = new ArrayList<Parametre>();

                //2.- Creació d'un paràmetre
                Parametre param1 = new Parametre();

                //2.1.- Creació de la clau del primer paràmetre
                JAXBElement<String> clau = new JAXBElement<String>(_ParametreClau_QNAME,
String.class, "text_horiz");
                param1.setClau(clau);
                //2.2.- Creació del valor de la clau (un stringSegell)
                StringSegell stringSegell = new StringSegell();

                //2.2.1.- carrega de les dades del StringSegell
                JAXBElement<String> text = new JAXBElement<String>(_StringSegellText_QNAME,
String.class, "Prova test Horitzontal");
                stringSegell.setText(text);

                //2.2.2.- Carrega del valor de la clau
                JAXBElement<StringSegell> valor = new
JAXBElement<StringSegell>(_ParametreValorString_QNAME, StringSegell.class, stringSegell);
                param1.setValorString(valor);

                //3.- carrega del paràmetre
                params.add(param1);
                //Crida al servei
                ResultSTD result = services.composarPDFRemot(config, nomFitxerEntrada, nomPlantilla,
porcentatgeEscalat,
                        posicioEscalatX, posicioEscalatY, params, nomFitxerSortida, generarCSV);

                JAXBElement<byte[]> out = result.getArxiu();

                if (out.getValue() != null) {
                        FileOutputStream fos =
                                new FileOutputStream(new
File("C:\\tmp\\STD\\ctti\\cscanigo\\documents\\composicio\\out\\prueba_composar_remot_OUT.pdf"));
                        fos.write(out.getValue());
                        fos.flush();
                        fos.close();
                }

                System.out.println("[TestComposarPDFRemot] Resultat: " +
result.getKey().getValue());
                System.out.println("[TestComposarPDFRemot] MissatgeError: " +
result.getMissatgeError().getValue());
                System.out.println("[TestComposarPDFRemot] TimeStamp: " +
result.getTimeStamp().getValue());
                System.out.println("[TestComposarPDFRemot] Estat: " +
result.getStatus().intValue());
                System.out.println("[TestComposarPDFRemot] Final");

        } catch (Exception e) {
                System.out.println("ERROR en el Test del Servei de composició de PDF (Arxius)");
```

```
                e.printStackTrace();
        }
}
```

```
@Test
public void TestComposarPDFStream() {

        try {

                System.out.println("[TestComposarPDFStream] Inici");
                //Constructor de objectes bàsics
                cat.gencat.ctti.canigo.eforms.services.objects.xsd.ObjectFactory objectFactory =
                        new cat.gencat.ctti.canigo.eforms.services.objects.xsd.ObjectFactory();

                String nomPlantilla = "plantilla_marge26_tot";
                Float percentatgeEscalat = new Float(1);
                Float posicioEscalatX = new Float(5);
                Float posicioEscalatY = new Float(5);
                boolean generarCSV = false;

                ConfigCall config = new ConfigCall();

                JAXBElement<String> ambit = objectFactory.createConfigCallAmbit(ambitProves);
                JAXBElement<String> aplicacio =
objectFactory.createConfigCallAmbit(aplicacioProves);

                config.setAmbit(ambit);
                config.setAplicacio(aplicacio);

                File fitxerEntrada = new
File("C:\\tmp\\STD\\ctti\\cscanigo\\documents\\composicio\\in\\prueba_composar.pdf");
                InputStream inStream = new FileInputStream(fitxerEntrada);

                ByteArrayOutputStream buffer = new ByteArrayOutputStream();
                int nRead;
                byte[] data = new byte[1024];
                while ((nRead = inStream.read(data, 0, data.length)) != -1) {
                  buffer.write(data, 0, nRead);
                }
                buffer.flush();

                byte[] inputStreamFile = buffer.toByteArray();

                //carrega de paràmetres de composició
                //1.- Creació llista de paràmetres
                List<Parametre> params = new ArrayList<Parametre>();
                //TODO BORRAR

                //2.- Creació d'un paràmetre
                Parametre param1 = new Parametre();

                //2.1.- Creació de la clau del primer paràmetre
                JAXBElement<String> clau = new JAXBElement<String>(_ParametreClau_QNAME,
String.class, "text_horiz");
                param1.setClau(clau);

                //2.2.- Creació del valor de la clau (un stringSegell)
                StringSegell stringSegell = new StringSegell();

                //2.2.1.- carrega de les dades del StringSegell
                JAXBElement<String> text = new JAXBElement<String>(_StringSegellText_QNAME,
String.class, "Prova test Horitzontal");
                stringSegell.setText(text);

                //2.2.2.- Carrega del valor de la clau
                JAXBElement<StringSegell> valor = new
JAXBElement<StringSegell>(_ParametreValorString_QNAME, StringSegell.class, stringSegell);
                param1.setValorString(valor);

                //3.- carrega del paràmetre
                params.add(param1);

                //Crida al servei
                ResultSTD result = services.composarPDFStream(config, inputStreamFile, nomPlantilla,
percentatgeEscalat,
                                posicioEscalatX, posicioEscalatY, params, generarCSV);
```

Creació Client CXF per l'STD

```
                JAXBElement<byte[]> out = result.getArxiu();

                if (out.getValue() != null) {
                        FileOutputStream fos =
                                new FileOutputStream(new
File("C:\\tmp\\STD\\ctti\\cscanigo\\documents\\composicio\\out\\prueba_composar_stream_OUT.pdf"));
                        fos.write(out.getValue());
                        fos.flush();
                        fos.close();
                }

                System.out.println("[TestComposarPDFStream] Resultat: " +
result.getKey().getValue());
                System.out.println("[TestComposarPDFStream] MissatgeError: " +
result.getMissatgeError().getValue());
                System.out.println("[TestComposarPDFStream] TimeStamp: " +
result.getTimeStamp().getValue());
                System.out.println("[TestComposarPDFStream] Estat: " +
result.getStatus().intValue());
                System.out.println("[TestComposarPDFStream] Final");

        } catch (Exception e) {
                System.out.println("ERROR en el Test del Servei de composició de PDF (Arxius)");
                e.printStackTrace();
        }
}
```

```
@Test
public void testConvertirWordDOCAPDF() {

        System.out.println("[testConvertirWordDOCAPDF] Inici");
        //Constructor de objectes bàsics
        cat.gencat.ctti.canigo.eforms.services.objects.xsd.ObjectFactory objectFactory =
                new cat.gencat.ctti.canigo.eforms.services.objects.xsd.ObjectFactory();

        ConfigCall config = new ConfigCall();

        JAXBElement<String> ambit = objectFactory.createConfigCallAmbit(ambitProves);
        JAXBElement<String> aplicacio = objectFactory.createConfigCallAmbit(aplicacioProves);

        config.setAmbit(ambit);
        config.setAplicacio(aplicacio);

        ConfigServiceSTD configSTD = new ConfigServiceSTD();

        JAXBElement<String> fitxerEntrada =
objectFactory.createConfigServiceSTDNomFitxerEntrada("prueba_doc.doc");
        configSTD.setNomFitxerEntrada(fitxerEntrada);

        JAXBElement<String> formatoEntrada =
objectFactory.createConfigServiceSTDFormatoEntrada("doc");
        configSTD.setFormatoEntrada(formatoEntrada);

        try{
                //Crida al servei
                ResultSTD result = services.convertirPDF(config, configSTD);

                JAXBElement<byte[]> out = result.getArxiu();
                if (out.getValue() != null) {
                        FileOutputStream fos =
                                new FileOutputStream(new
File("C:\\tmp\\STD\\ctti\\cscanigo\\documents\\office\\out\\prueba_convertir_doc_out.pdf"));
                        fos.write(out.getValue());
                        fos.flush();
                        fos.close();
                }

                System.out.println("[testConvertirWordDOCAPDF] Resultat: " +
result.getKey().getValue());
                System.out.println("[testConvertirWordDOCAPDF] MissatgeError: " +
result.getMissatgeError().getValue());
                System.out.println("[testConvertirWordDOCAPDF] Estat: " +
result.getStatus().intValue());
                System.out.println("[testConvertirWordDOCAPDF] Final");

        } catch (Exception e) {
                e.printStackTrace();
```

```
        }
}
```

```
@Test
public void TestConvertirComposarPDF() {

        try {
                System.out.println("[TestConvertirComposarPDF] Inici");
                //Constructor de objectes bàsics
                cat.gencat.ctti.canigo.eforms.services.objects.xsd.ObjectFactory objectFactory =
                        new cat.gencat.ctti.canigo.eforms.services.objects.xsd.ObjectFactory();

                String nomFitxerEntrada = "prueba_doc.doc";
                String formatoEntrada = "doc";
                String nomPlantilla = "plantilla_marge26_tot";
                String nomFitxerSortida="prueba_convertir_composar_out.pdf";
                Float porcentatgeEscalat = new Float(1);
                Float posicioEscalatX = new Float(5);
                Float posicioEscalatY = new Float(5);
                boolean generarCSV = true;

                ConfigCall config = new ConfigCall();

                JAXBElement<String> ambit = objectFactory.createConfigCallAmbit(ambitProves);
                JAXBElement<String> aplicacio =
objectFactory.createConfigCallAmbit(aplicacioProves);

                config.setAmbit(ambit);
                config.setAplicacio(aplicacio);

                //carrega de paràmetres de composició
                //1.- Creació llista de paràmetres
                List<Parametre> params = new ArrayList<Parametre>();

                //2.- Creació d'un paràmetre
                Parametre param1 = new Parametre();

                //2.1.- Creació de la clau del primer paràmetre
                JAXBElement<String> clau = new JAXBElement<String>(_ParametreClau_QNAME,
String.class, "text_horiz");
                param1.setClau(clau);

                //2.2.- Creació del valor de la clau (un stringSegell)
                StringSegell stringSegell = new StringSegell();

                //2.2.1.- carrega de les dades del StringSegell
                JAXBElement<String> text = new JAXBElement<String>(_StringSegellText_QNAME,
String.class, "Prova test Horitzontal");
                stringSegell.setText(text);

                //2.2.2.- Carrega del valor de la clau
                JAXBElement<StringSegell> valor = new
JAXBElement<StringSegell>(_ParametreValorString_QNAME, StringSegell.class, stringSegell);
                param1.setValorString(valor);

                //3.- carrega del paràmetre
                params.add(param1);

                //Crida al servei
                ResultSTD result = services.convertirComposarPDF(config, nomFitxerEntrada,
formatoEntrada, nomPlantilla,
                        porcentatgeEscalat,      posicioEscalatX, posicioEscalatY, params,
nomFitxerSortida, generarCSV);

                JAXBElement<byte[]> out = result.getArxiu();

                if (out.getValue() != null) {
                        FileOutputStream fos =
                                new FileOutputStream(new
File("C:\\STD\\documents\\out\\prueba_convertir_composar_OUT.pdf"));
                        fos.write(out.getValue());
                        fos.flush();
                        fos.close();
                }
                System.out.println("[TestConvertirComposarPDF] Resultat: " +
result.getKey().getValue());
                System.out.println("[TestConvertirComposarPDF] MissatgeError: " +
result.getMissatgeError().getValue());
```

Creació Client CXF per l'STD

```
                System.out.println("[TestConvertirComposarPDF] Estat: " +
result.getStatus().intValue());
                System.out.println("[TestConvertirComposarPDF] Final");

        } catch (Exception e) {
                e.printStackTrace();
        }
}
```

Finalment per qualsevol dubte o problema que pugui sorgir es pot fer una consulta al CS Canigó a la bústia oficina-tecnica.canigo.ctti@gencat.net o bé mitjançant una petició de suport a l'eina JIRA del CTTI en http ://cstd.ctti.gencat.cat/jiracstd.