1 INTRODUCCIÓN

En este documento se expone toda la información técnica para entender el desarrollo de la aplicación Gest-OLI, con la finalidad de que cualquier persona con conocimientos técnicos y de programación en J2EE pueda realizar cualquier modificación/mejora de la aplicación e implantarla en producción.

Dividiremos el documento en cuatro partes bien diferenciadas:

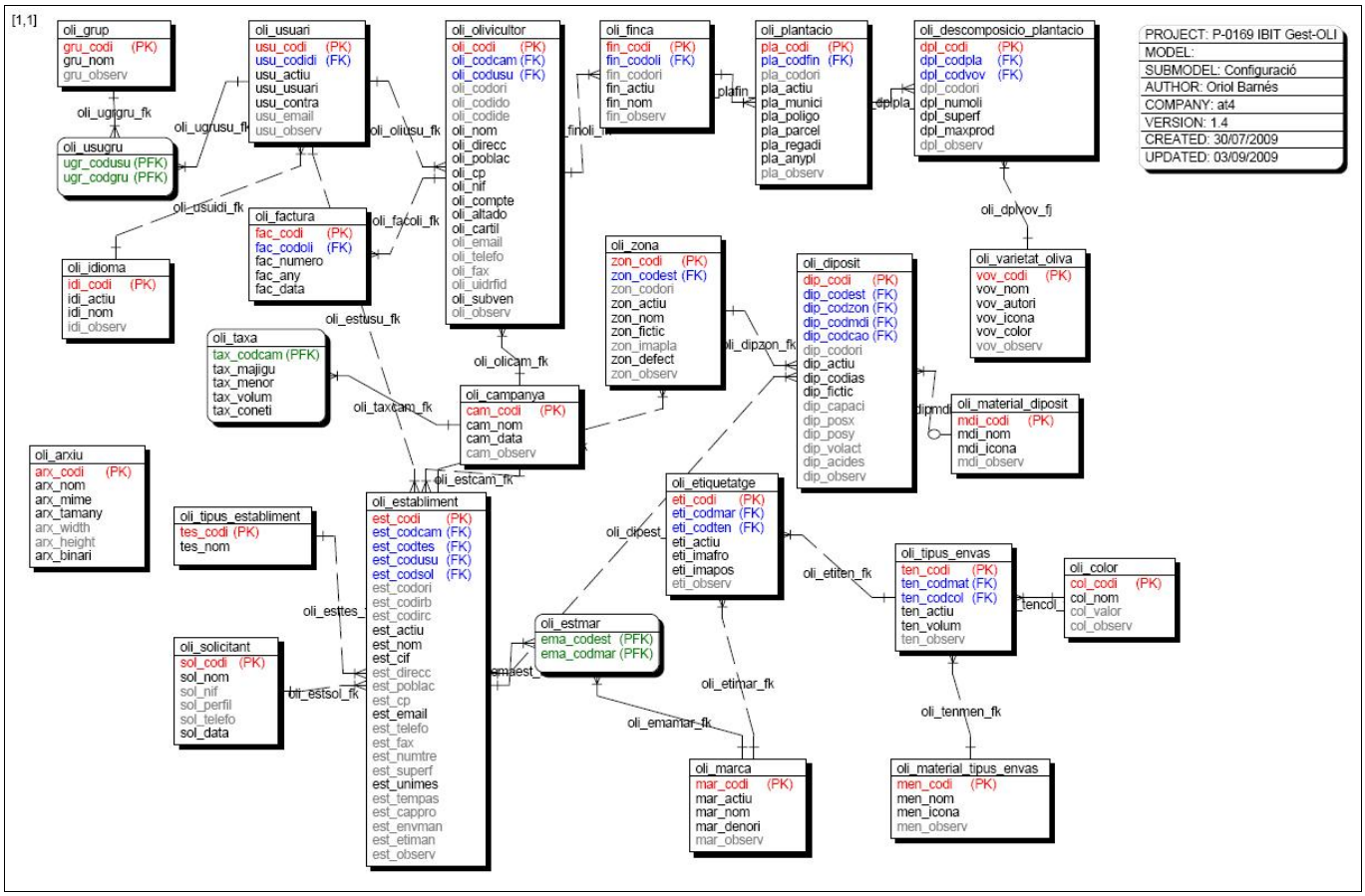
- Diseño de base de datos
- Modelo de datos
- Desarrollo
- Control de calidad
- Implantación

2 DISEÑO DE BASE DE DATOS

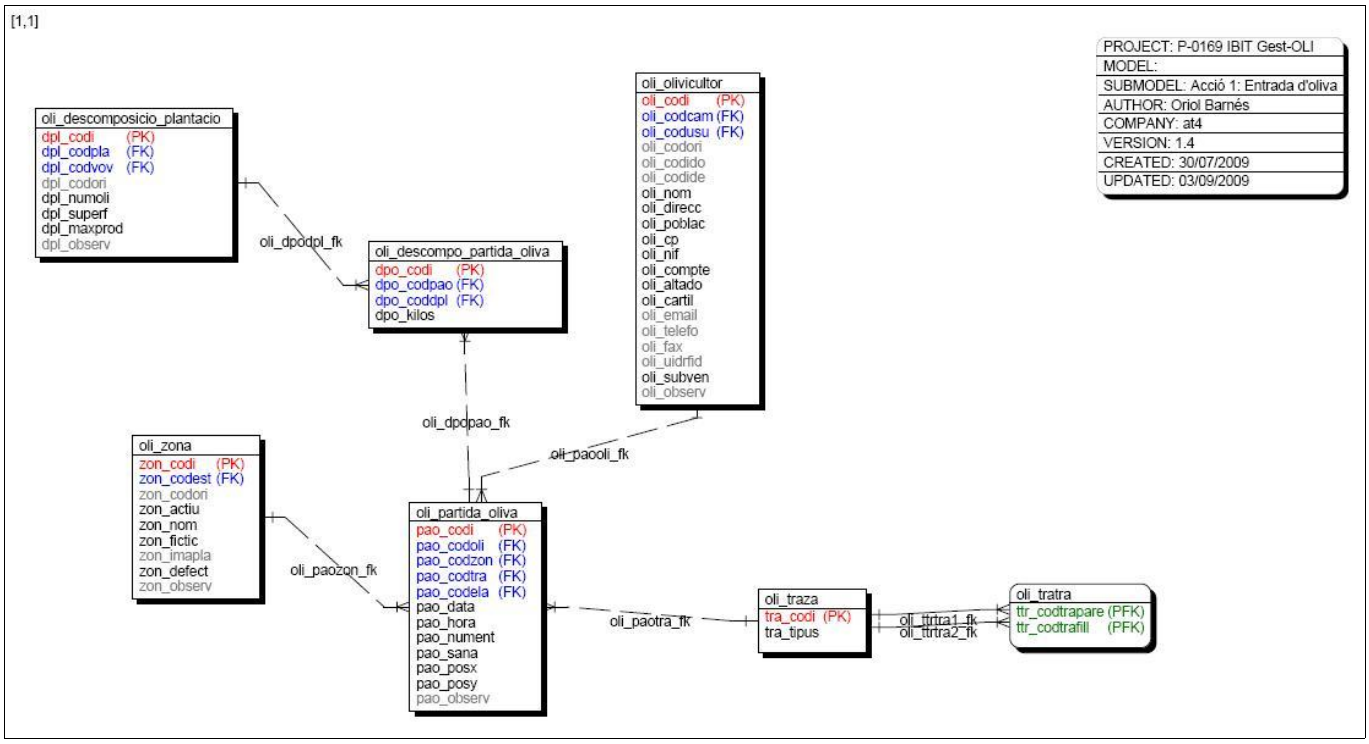
Se ha utilizado **PostgreSQL 8.2.7** como motor de base de datos e **Hibernate2** como herramienta de mapeo objeto-relacional para la plataforma Java, que facilita el mapeo de atributos entre una base de datos relacional tradicional y el modelo de objetos de una aplicación, mediante archivos declarativos (XML) que permiten establecer estas relaciones.

Los esquemas ERD correspondientes con el diseño son los siguientes:

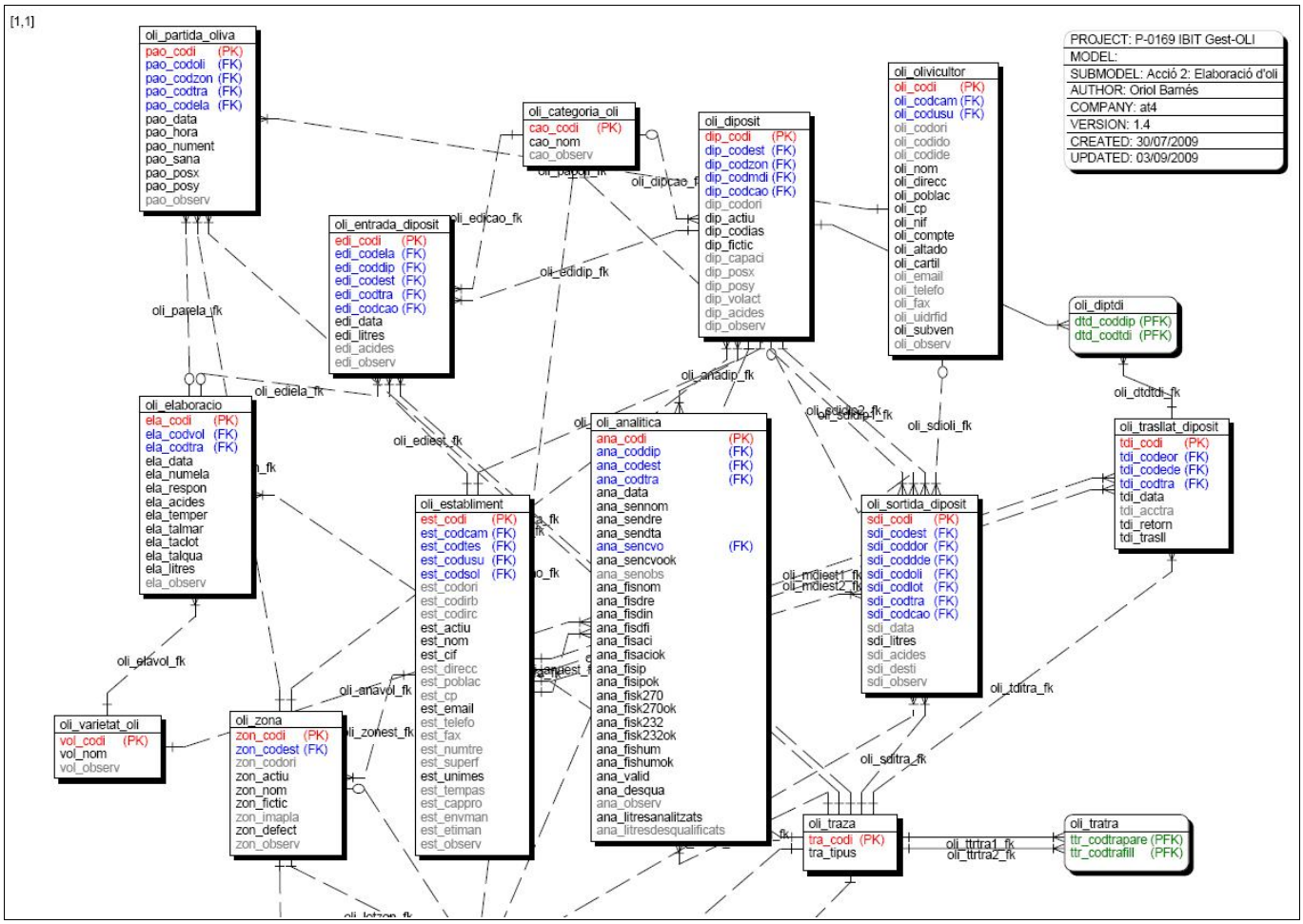
at4.net internet y comunicación . Santiago Rusiñol, 8. primero – Palma de Mallorca
www.at4.net tel. +34 971 22 88 70 – fax. +34 971 22 88 71 – info@at4.net

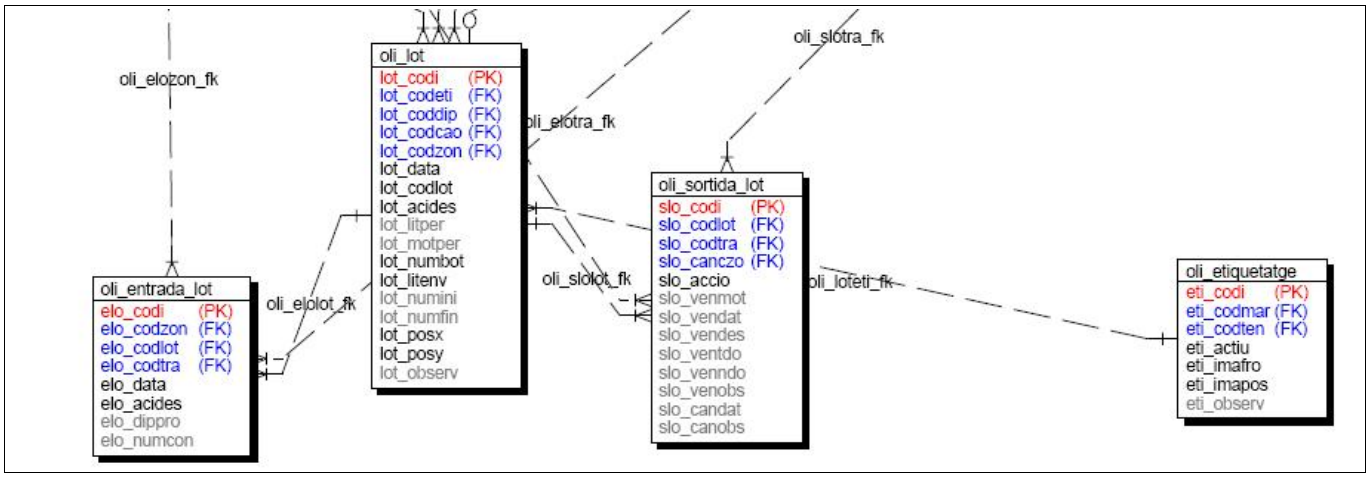


at4.net internet y comunicación . Santiago Rusiñol, 8. primero – Palma de Mallorca
www.at4.net tel. +34 971 22 88 70 – fax. +34 971 22 88 71 – info@at4.net

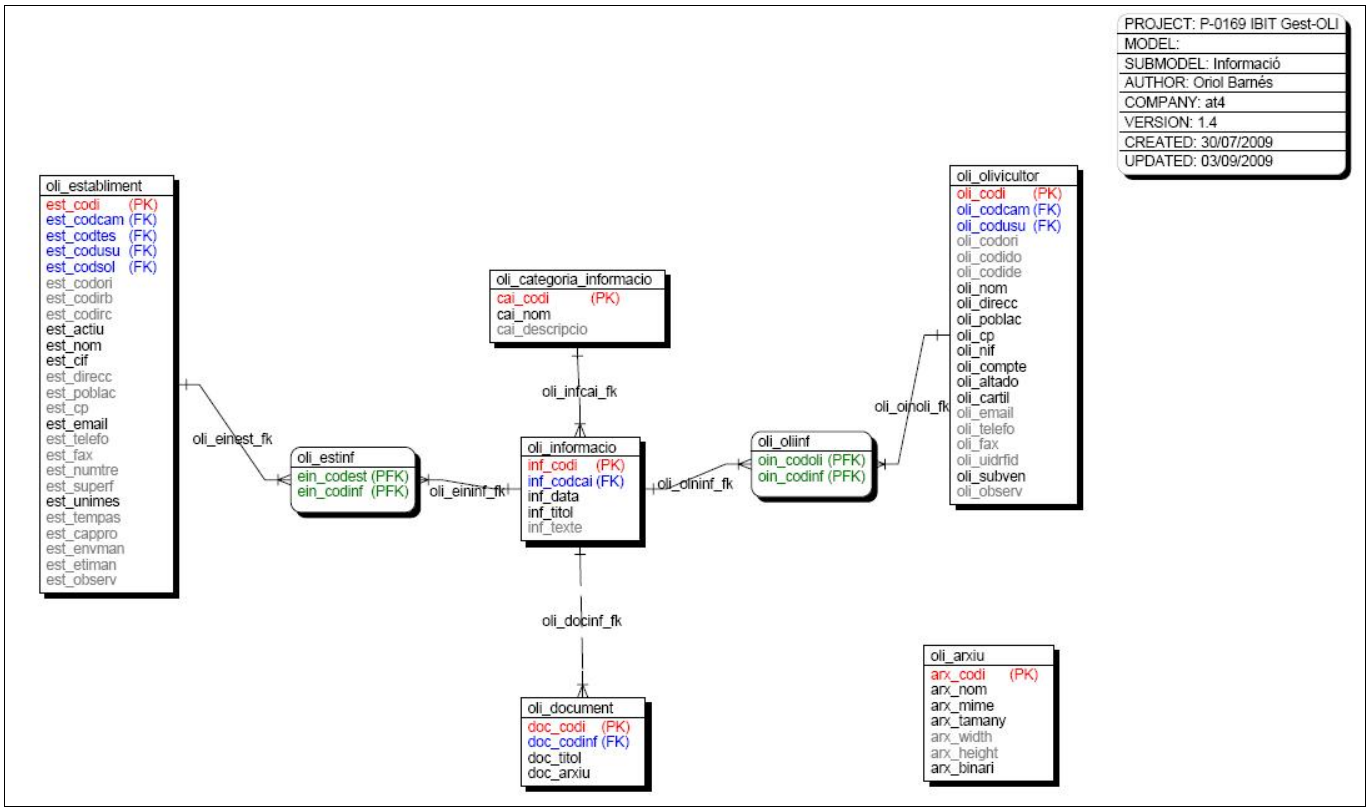


at4.net internet y comunicación . Santiago Rusiñol, 8. primero – Palma de Mallorca
www.at4.net tel. +34 971 22 88 70 – fax. +34 971 22 88 71 – info@at4.net





at4.net internet y comunicación . Santiago Rusiñol, 8. primero – Palma de Mallorca
www.at4.net tel. +34 971 22 88 70 – fax. +34 971 22 88 71 – info@at4.net

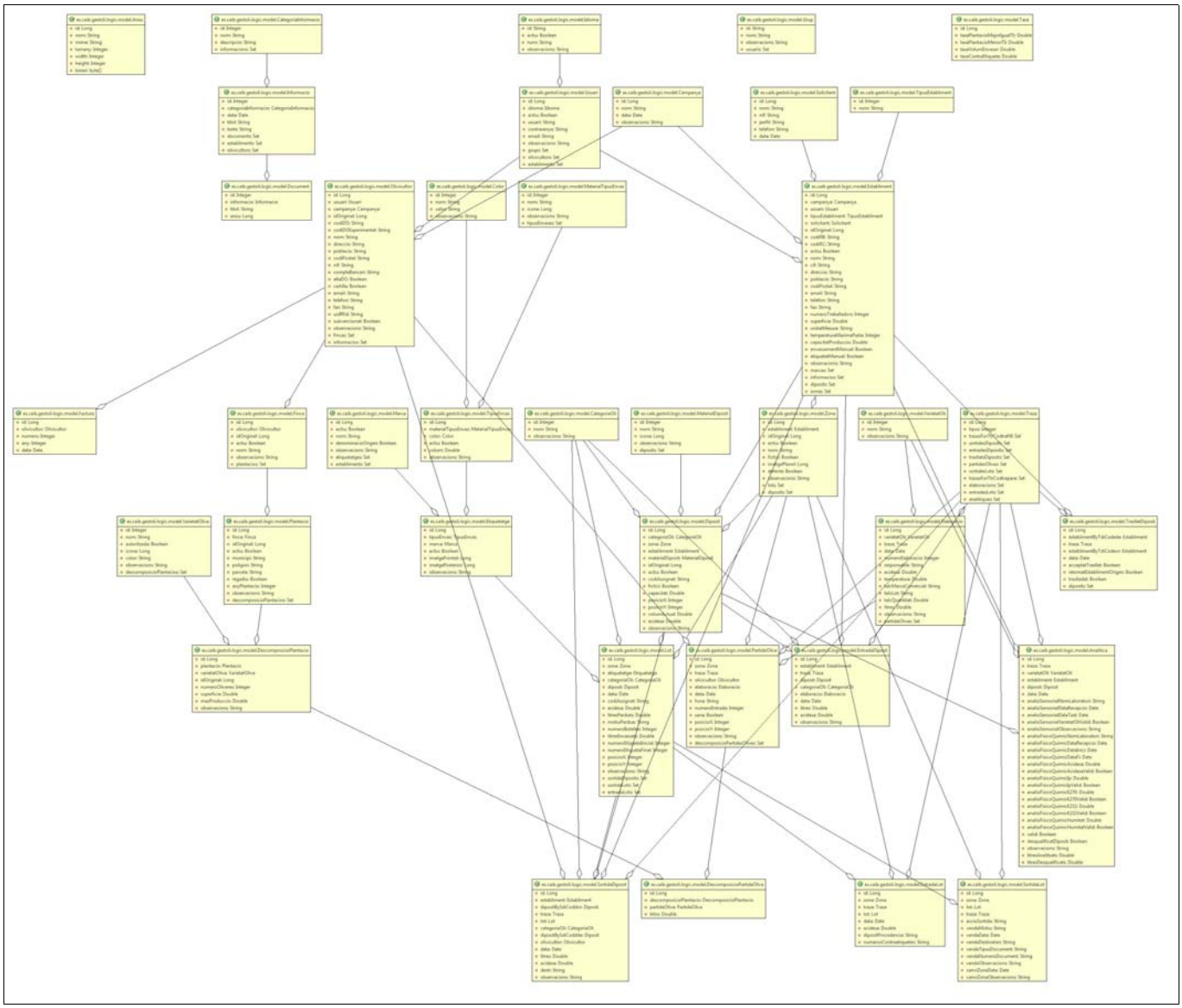


El ERD en formato PDF se puede encontrar en *./docs/GEST-OLI - Documentación Técnica - ERD.pdf*.

3 MODEL DE DATOS

A continuación mostramos el diagrama con el modelo de datos:

at4.net internet y comunicación . Santiago Rusiñol, 8. primero – Palma de Mallorca
www.at4.net tel. +34 971 22 88 70 – fax. +34 971 22 88 71 – info@at4.net



El modelo en formato PDF se puede encontrar en *./docs/GEST-OLI - Documentación Técnica - Modelo de Datos.pdf*.

at4.net internet y comunicación . Santiago Rusiñol, 8. primero – Palma de Mallorca
www.at4.net tel. +34 971 22 88 70 – fax. +34 971 22 88 71 – info@at4.net

4 DESARROLLO

Para el desarrollo de la aplicación se ha utilizado **Eclipse** como entorno de desarrollo integrado (IDE). Todo el código fuente reside en el repositorio CVS de IBIT, concretamente en el servidor **cvs.ibit.org** y módulo **gestoli**. Dentro del código fuente, se encuentran los archivos de configuración del Eclipse (.project y .classpath) con los cuales se montaría automáticamente el proyecto de la siguiente manera:

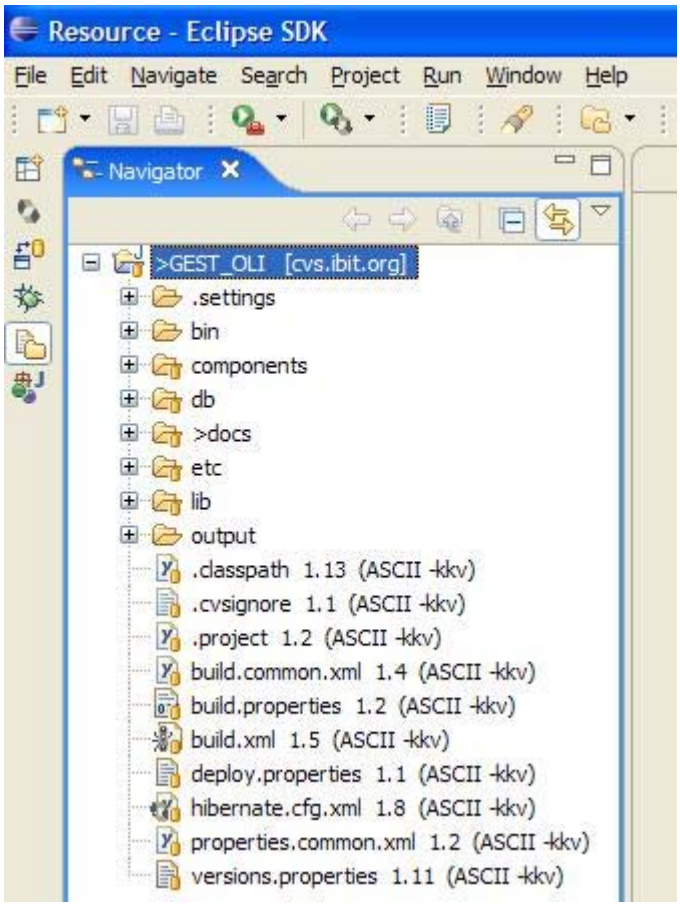


Ilustración 1: Proyecto en Eclipse

Las clases Java se compilan con JDK1.4 y la codificación de los archivos es UTF-8. Se ha utilizado el framework de java Spring[1] para la construcción y validación de formularios.

A continuación se explicarán cada uno de los distintos archivos, carpetas, packages, etc. que forman la aplicación.

1 Spring: http://www.springsource.org/

4.1 root

- *.classpath*: archivo de configuración del classpath en Eclipse
- *.cvsignore*: archivo ignore de CVS para esta carpeta raíz, que actualmente solo tiene tres entradas:

  output

  bin

  .settings

  que son los archivos y carpetas que se generan dinámicamente con Eclipse y con el ant.
- *.project*: archivo de configuración del proyecto en Eclipse
- *build.common.xml*: archivo complementario al build.xml
- *build.properties*: archivo de propiedades para el build.xml
- *build.xml*: archivo principal de la ejecución con distintas operaciones a ejecutar. La ejecución debe hacerse directamente sobre una ventana terminal y no desde Eclipse. Entre las operaciones que se pueden ejecutar, destacan:

  ant clean → borrar todos los archivos .class generados (borrado de la carpeta "output")

  ant make → compila todos los archivos Java, crea todas las interfaces e implementaciones de los EJBs y finalmente genera el archivo gestoli.ear que contendrá toda la aplicación

  ant deploy.remote → copia el archivo gestoli.ear dentro de la carpeta "deploy" del JBoss
- *deploy.properties*: archivo complementario al build.xml
- *hibernate.cfg.xml*: archivo de configuración del mapeo Hibernate. Este archivo solo es útil para utilización del plugin HibernateTool y así poder ejecutar sentencias HQL
- *properties.common.xml*: archivo complementario al build.xml
- *versions.properties*: archivo complementario al build.xml. Este archivo sirve para anotar las distintas versiones de archivos jar

4.2 components

En este directorio residen los 4 componentes distintos que forman la aplicación.

at4.net internet y comunicación . Santiago Rusiñol, 8. primero – Palma de Mallorca
www.at4.net tel. +34 971 22 88 70 – fax. +34 971 22 88 71 – info@at4.net

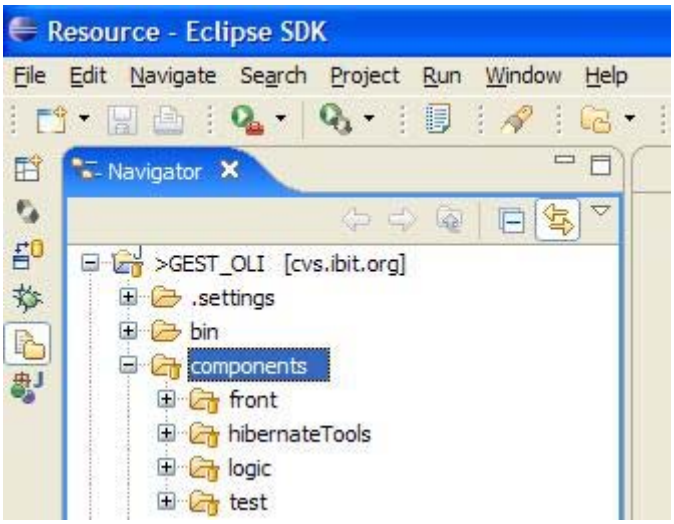


Ilustración 2: directorio components

A continuación se detalla uno a uno por separado.

4.2.1 front

Contiene todos los archivos necesarios para la capa de negocio y presentación de la aplicación. En la raíz del directorio, se encuentran los archivos *build.properties*, *build.xml* y *deploy.properties* necesarios para la compilación del componente *front*. A continuación se detallan los distintos subdirectorios.

4.2.1.1 front/etc

Aquí se encuentran la mayoría de archivos de configuración, propiedades, idiomas, etc.

En la carpeta *applet* hay los archivos "drivers" para la configuración del lector RFID[2].

En la carpeta *merge* se encuentran los archivos a partir de los cuales se genera el archivo *web.xml* que define la aplicación gestoli.war.

En la carpeta *resources* se encuentran los archivos de literales multi-idioma.

A continuación se detallan un poco más exhaustivamente los archivos que se encuentran en la raíz de *front/etc*:

- app.properties → propiedades con constantes que se inyectan en las clases configuradas por Spring

2 Lector RFID de Kimaldi: http://www.kimaldi.com/productos/sistemas_rfid

at4.net internet y comunicación . Santiago Rusiñol, 8. primero – Palma de Mallorca
www.at4.net tel. +34 971 22 88 70 – fax. +34 971 22 88 71 – info@at4.net

- applicationContext.xml → archivo de configuración de Spring
- app-servlet.xml → archivo de configuración de Spring. En este archivo se configuran los *interceptors* y los *controllers* que son llamados en cada una de las distintas *URLs*
- decorators.xml → archivo de configuración de sitemesh (indica la plantilla a utilizar)
- dwr.xml → archivo de configuración de *DWR*[3] donde se configuran los distintos servicios *AJAX*[4] que son llamados desde las páginas *JSP*.
- hibernate.cfg.xml → archivo de configuración del mapeo de Hibernate
- sitemesh.xml → archivo de configuración de sitemesh

4.2.1.2 front/httpdocs

Aquí se encuentran todos los archivos pertenecientes a la capa de presentación, tales como hojas de estilo, javascripts, imágenes, JSPs, etc.

Hay que tener especial atención en la carpeta *httpdocs/applet*, ya que aquí se encuentra el applet de interacción con el lector RFID. Cuando se hace el *ant make* se depositan aquí las clases compiladas del applet.

4.2.1.3 front/src

Aquí se encuentran todas las clases Java de la capa de negocio.

A continuación se detallan un poco más exhaustivamente los distintos packages que se encuentran en este directorio:

- applet/rfid → clases del applet de interacción con el lector RFID
- cron → la clase *TrasllatDipositJob* realiza el envío desatendido de depósitos entre un establecimiento y otro. Para la ejecución de este envío, se utiliza la librería Quartz[5] cuya configuración se encuentra en el archivo *front/etc/applicationContext.xml*
- dwr → clases que definen los servicios AJAX
- spring → clases *Command*, *Controller*, *Delegate* y *Validator* de cada una de las distintas peticiones al servidor. Estas clases implementan el framework Spring.

3 Direct Web Remoting: http://directwebremoting.org/dwr/index.html
4 Asynchronous JavaScript and XHTML: http://en.wikipedia.org/wiki/Ajax_(programming)
5 Quartz Enterprise Job Scheduler: http://www.opensymphony.com/quartz/

- spring/interceptor → clase interceptora que se ejecuta siempre en primer lugar, en todas y cada una de las peticiones al servidor
- spring/views → clases que generan informes en formato PDF
- util → clases de utilidades varias

4.2.2 hibernateTools

Este componente se utiliza únicamente para la creación de las clases de modelo a partir de las tablas de base de datos. Se necesita tener instalado el plugin HibernateTools[6] y con los archivos *hibernate.reveng.xml* y *src/hibernate.cfg.xml* y *src/OliDoLibCustomStrategy.java* ya se generan automáticamente todas las clases de modelo y su archivo de mapeo correspondiente *.hbm.xml*.

4.2.3 logic

Contiene todos los archivos necesarios para la capa de persistencia de la aplicación. En la raíz del directorio, se encuentran los archivos *build.xml* y *deploy.properties* necesarios para la compilación del componente *logic*. A continuación se detallan los distintos subdirectorios.

4.2.3.1 logic/src

Aquí se encuentran todas las clases Java de la capa de persistencia.

A continuación se detallan un poco más exhaustivamente los distintos packages que se encuentran en este directorio:

- dao → clases que a través de Hibernate realizan todas las acciones sobre la base de datos, tales como consultas, altas, modificaciones y borrados.
- ejb → clases EJBs de sesión Stateless. Éstas hacen de puente entre las clases de negocio y las de persistencia, para así aislar al máximo las distintas capas.
- exception → clase específica para excepciones
- model → clases que modelan cada una de las distintas entidades de base de datos (y las relaciones entre ellas). Para cada clase, hay también su archivo de mapeo correspondiente .hbm.xml de Hibernate.
- resources → aquí se encuentran los distintos iconos de tipos de oliva, depósitos y envases. Aquí también se encuentran las plantillas JRXML[7] de los distintos informes/documentos que generan un archivo PDF.

6 Hibernate Tools for Eclipse and Ant: https://www.hibernate.org/255.html
7 Plantilla XML de JasperReports: http://jasperreports.sourceforge.net/

at4.net internet y comunicación . Santiago Rusiñol, 8. primero – Palma de Mallorca
www.at4.net tel. +34 971 22 88 70 – fax. +34 971 22 88 71 – info@at4.net

- util → clases de utilidades varias

4.2.4 test

Este componente contiene las distintas clases java utilizadas para una batería de test específico.

Este apartado ya se comentará más adelante en la sección de CONTROL DE CALIDAD.

4.3 db

En este directorio hay varios archivos relacionados con la base de datos y la conexión de la aplicación con ella.

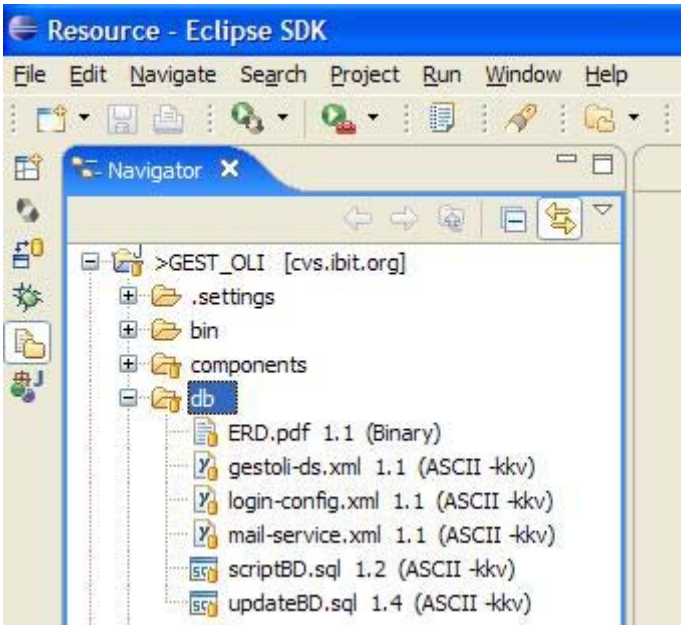


Ilustración 3: directorio db

A continuación se detalla cada uno de los archivos por separado:

- *ERD.pdf*: archivo pdf con el esquema ERD de la base de datos
- *gestoli-ds.xml*: archivo de configuración del *datasource*. Este archivo se debe copiar en la carpeta *deploy* del JBoss
- *login-config.xml*: el archivo *login-config.xml* de la carpeta *conf* del JBoss, debe tener esta *application-policy*

- *mail-service.xml*: archivo de configuración del servicio de envío de emails. Este archivo contiene el `mbean` con nombre `name="jboss:service=MailGestoli"`, el cual tiene que estar configurado en el archivo *mail-service.xml* de la carpeta *deploy* del JBoss
- scriptBD.sql: script SQL con la generación de la base de datos inicial
- updateBD.sql: script SQL con las modificaciones sobre la base de datos inicial

4.4 docs

En este directorio residen todos los documentos y manuales de la aplicación.

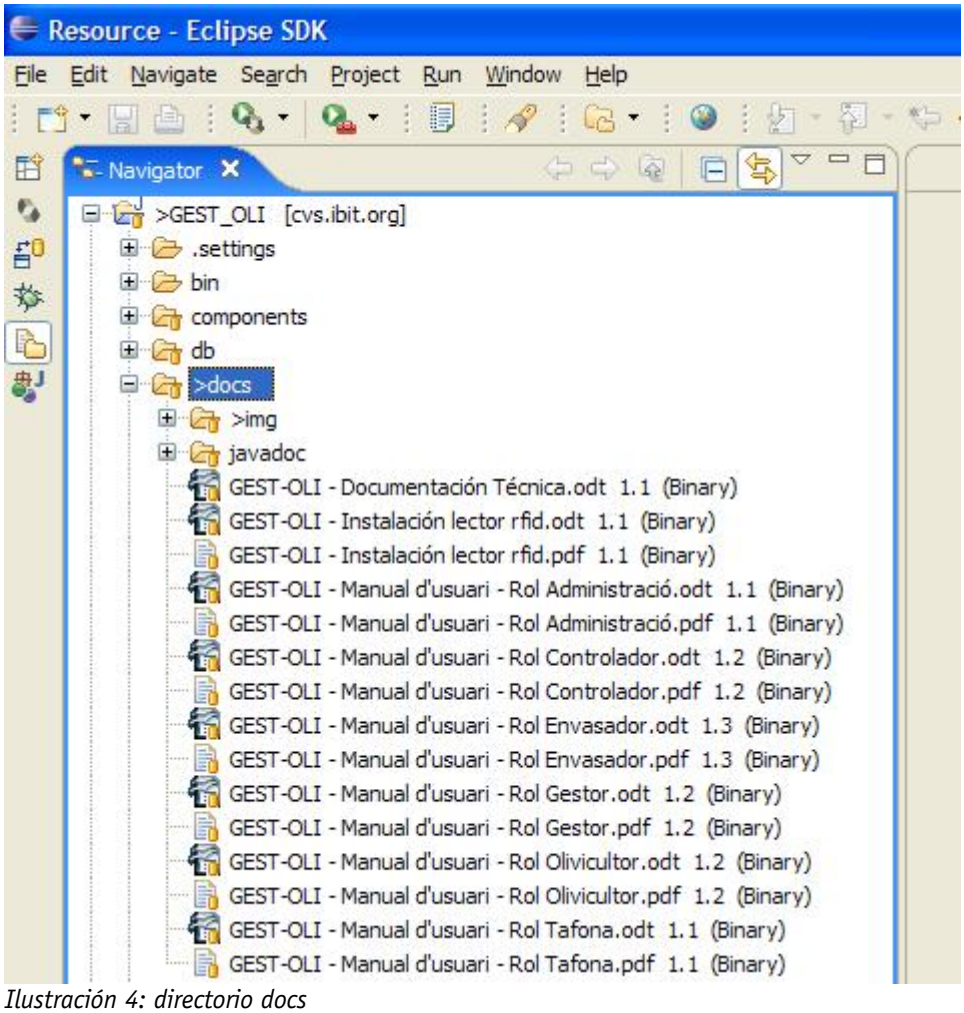


Ilustración 4: directorio docs

A continuación se detalla el contenido de este directorio:

- *img*: archivos con las capturas de pantalla de los distintos documentos/manuales

at4.net internet y comunicación . Santiago Rusiñol, 8. primero – Palma de Mallorca
www.at4.net tel. +34 971 22 88 70 – fax. +34 971 22 88 71 – info@at4.net

- *javadoc*: javadoc en formato HTML de todas las clases JAVA de la aplicación
- *GEST-OLI - Documentación Técnica.odt*: este documento
- *GEST-OLI - Documentación Técnica – ERD.pdf*: ERD de la aplicación
- *GEST-OLI - Documentación Técnica - Modelo de Datos.pdf*: modelo de datos de la aplicación
- *GEST-OLI - Instalación lector rfid.odt*: manual de instalación del lector RFID
- *GEST-OLI - Manual d'usuari - Rol Administració.odt*: manual de usuario con rol de Administración
- *GEST-OLI - Manual d'usuari - Rol Controlador.odt*: manual de usuario con rol de Controlador
- *GEST-OLI - Manual d'usuari - Rol Envasador.odt*: manual de usuario con rol de Envasador
- *GEST-OLI - Manual d'usuari - Rol Gestor.odt*: manual de usuario con rol de Gestor
- *GEST-OLI - Manual d'usuari - Rol Olivicultor.odt*: manual de usuario con rol de Olivicultor
- *GEST-OLI - Manual d'usuari - Rol Tafona.odt*: manual de usuario con rol de Tafona (almazara)

4.5 etc

En este directorio residen varios archivos de configuración del EAR.

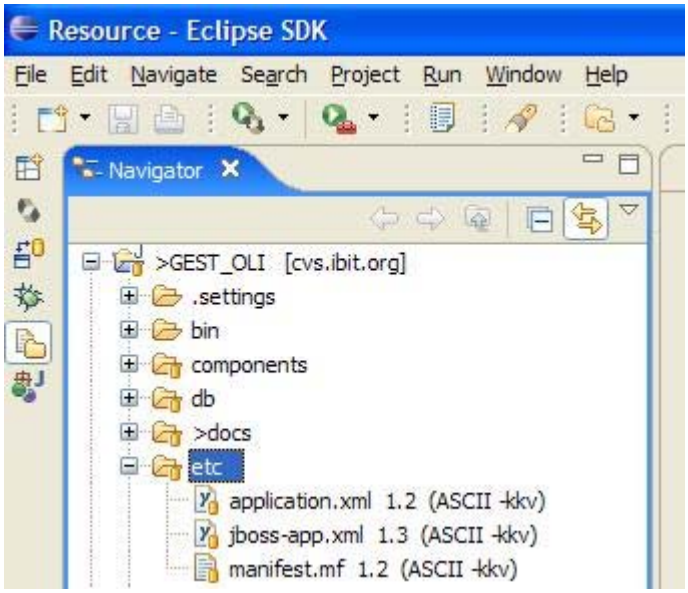


Ilustración 5: directorio etc

at4.net internet y comunicación . Santiago Rusiñol, 8. primero – Palma de Mallorca
www.at4.net tel. +34 971 22 88 70 – fax. +34 971 22 88 71 – info@at4.net

4.6 lib

En este directorio residen todas las librerías externas que utiliza la aplicación.

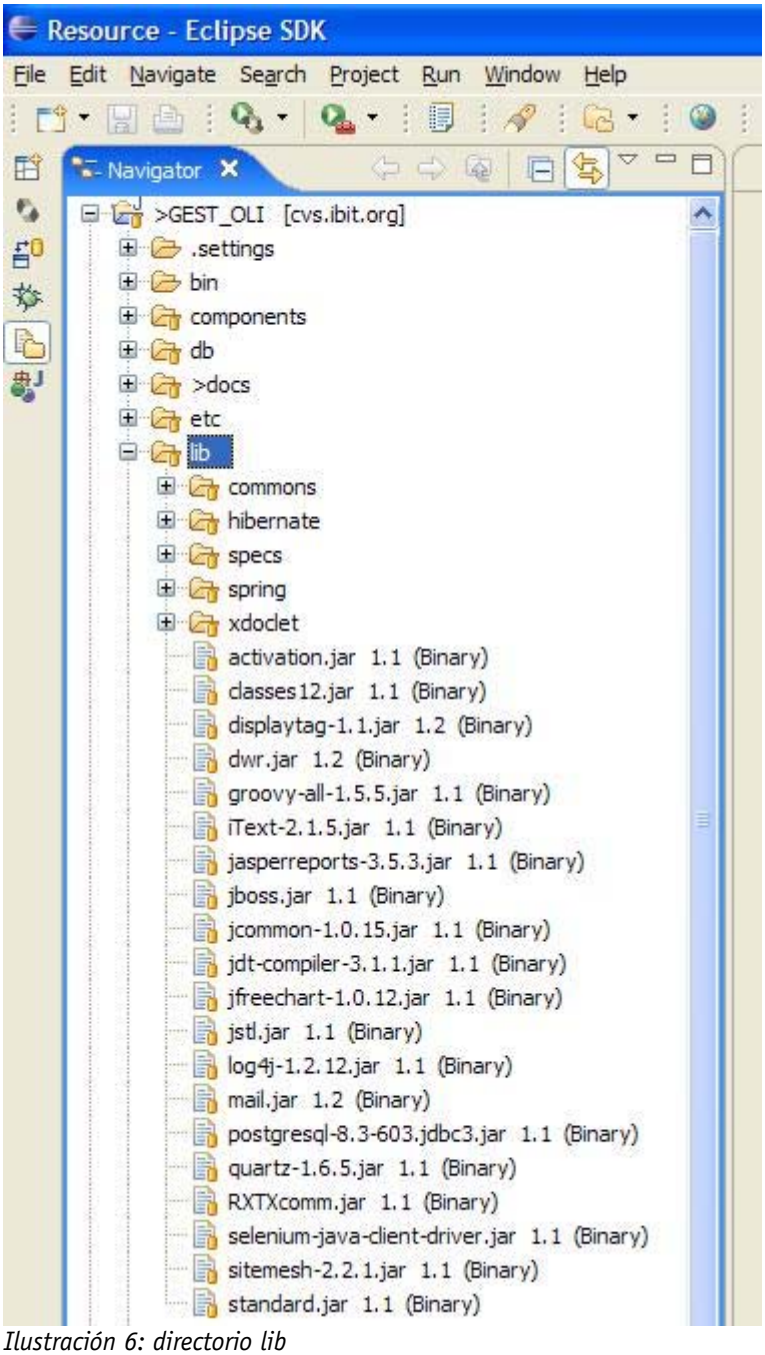


Ilustración 6: directorio lib

at4.net internet y comunicación . Santiago Rusiñol, 8. primero – Palma de Mallorca
www.at4.net tel. +34 971 22 88 70 – fax. +34 971 22 88 71 – info@at4.net

5 CONTROL DE CALIDAD

Para testear toda la aplicación se ha procedido a varios procedimientos, algunos automáticos y algunos otros más tradicionales. A continuación se detallan los distintos métodos.

5.1 Tests unitarios con Selenium

Selenium IDE[8] es un plugin de Firefox que pertenece al juego de herramientas SeleniumHQ[9], permite realizar juegos de pruebas sobre aplicaciones web. Para ello realiza la grabación de la acción seleccionada (navegación por una página) en un "script", el cual se puede editar y parametrizar para adaptarse a los diferentes casos, y lo que es más importante su ejecución se puede repetir tantas veces como se quiera (manual[10]).

Programas necesarios para la creación/ejecución de los tests unitarios:

- Selenium IDE Extension para Firefox: permite grabar y ejecutar scripts de acciones sobre el navegador (registra todo lo que estamos haciendo). En particular interesa la opción de exportar el script a código jUnit[11].
- Selenium RC: tiene el servidor Selenium y el driver Selenium para java. Se descomprime en cualquier carpeta.
- Eclipse: los scripts creados con el plugin para Firefox pueden ejecutarse directamente con jUnit desde Eclipse.

El juego de tests creados para esta ocasión se encuentran en *components/test*. A modo de ejemplo se muestra la clase UsuariTest.java que ha creado Selenium IDE:

```
package es.caib.gestoli.test;

import junit.framework.TestCase;
import com.thoughtworks.selenium.DefaultSelenium;
import com.thoughtworks.selenium.Selenium;
import com.thoughtworks.selenium.SeleniumException;



public class UsuariTest extends TestCase {
```

8 http://seleniumhq.org/download/
9 http://seleniumhq.org/
10 http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=seleniumIDE
11 http://www.junit.org/

at4.net internet y comunicación . Santiago Rusiñol, 8. primero – Palma de Mallorca
www.at4.net tel. +34 971 22 88 70 – fax. +34 971 22 88 71 – info@at4.net

```
	private Selenium selenium;


	public void setUp() throws Exception {
		selenium = new
DefaultSelenium("localhost",4444,"*firefox","http://gestor:gestor@gestoli.in.at4.net:8080/ges
toli/Inici.html?siteLanguage=ca");
		selenium.start();
	}

	public void testUsuariAlta() throws InterruptedException {
		selenium.open("/gestoli/Inici.html");
		selenium.click("link=Configuració");
		selenium.click("link=Usuari");
		selenium.waitForPageToLoad("30000");
		selenium.click("enlaceCrearForm");
		selenium.waitForPageToLoad("30000");
		selenium.type("usuari", "GestorTest");
		selenium.type("contrasenya", "gestor");
		selenium.select("idiomaId", "label=Català");
		selenium.addSelection("id_grupsArray", "label=Consell Regulador - Gestor");
		selenium.click("moveRight");
		selenium.type("id_observacions", "Observacions gestor");
		selenium.click("enlaceGuardarForm");
		selenium.waitForPageToLoad("30000");
		assertTrue(selenium.isTextPresent("L'usuari s'ha creat amb èxit"));

	}


	public void testUsuariAltaYaExiste() throws InterruptedException {
		selenium.open("/gestoli/Inici.html");
		selenium.click("link=Configuració");
		selenium.click("link=Usuari");
		selenium.waitForPageToLoad("30000");
		selenium.click("enlaceCrearForm");
		selenium.waitForPageToLoad("30000");
		selenium.type("usuari", "GestorTest");
		selenium.type("contrasenya", "gestor");
		selenium.select("idiomaId", "label=Català");
		selenium.addSelection("id_grupsArray", "label=Consell Regulador - Gestor");
		selenium.click("moveRight");
		selenium.type("id_observacions", "Observacions gestor");
		selenium.click("enlaceGuardarForm");
		selenium.waitForPageToLoad("30000");
		assertTrue(selenium.isTextPresent("El nom d'usuari ja existeix"));

	}


	public void testUsuariBaja() throws InterruptedException {
		selenium.open("/gestoli/Inici.html");
		selenium.click("link=Configuració");
		selenium.click("link=Usuari");
		selenium.waitForPageToLoad("30000");

		// Busqueda
		String text=null;
		while (true){

			try{
				text=selenium.getText("link=GestorTest");
				break;
			}catch(SeleniumException e){
```


at4.net internet y comunicación . Santiago Rusiñol, 8. primero – Palma de Mallorca
www.at4.net tel. +34 971 22 88 70 – fax. +34 971 22 88 71 – info@at4.net

```
                        try{
                                if
(selenium.getAttribute("xpath=//div[@id='divSombra_0']/div[8]/span/a[last()]/@href").equals("
#")){
                                        break;
                                }
                        }catch(SeleniumException e2){
                                break;
                        }
                        selenium.click("//div[@id='divSombra_0']/div[8]/span/a[last()]");
                        selenium.waitForPageToLoad("30000");
                }

        }

        assertTrue(text!=null);

        // Borrado
        selenium.click("link=GestorTest");
        selenium.waitForPageToLoad("30000");
        selenium.click("enlaceBorrarForm");
        assertTrue(selenium.getConfirmation().matches("^Esborrar el registre[\\s\\S]
Aquesta acció no es pot desfer\\.$"));
        selenium.waitForPageToLoad("30000");

        selenium.click("link=Configuració");
        selenium.click("link=Usuari");
        selenium.waitForPageToLoad("30000");

        // Busqueda
        text=null;
        while (true){

                try{
                        text=selenium.getText("link=GestorTest");
                        break;
                }catch(SeleniumException e){
                        try{
                                if
(selenium.getAttribute("xpath=//div[@id='divSombra_0']/div[8]/span/a[last()]/@href").equals("
#")){
                                        break;
                                }
                        }catch(SeleniumException e2){
                                break;
                        }
                        selenium.click("//div[@id='divSombra_0']/div[8]/span/a[last()]");
                        selenium.waitForPageToLoad("30000");
                }

        }

        assertTrue(text==null);

    }


    public void tearDown() {
            selenium.stop();
    }
```


at4.net internet y comunicación . Santiago Rusiñol, 8. primero – Palma de Mallorca
www.at4.net tel. +34 971 22 88 70 – fax. +34 971 22 88 71 – info@at4.net

```
}
```

Los tests pueden asociarse a un macrocontenedor TestSuite de jUnit que tiene esta estructura:

```
package es.caib.gestoli.test;
import junit.framework.Test;
import junit.framework.TestSuite;
import junit.textui.TestRunner;


      public class VariosTestSuite extends TestSuite {

      public VariosTestSuite(String name) {
        super(name);
      }

      public static void main(String[] args) {
        TestRunner.run(suite());
      }

      public static Test suite() {
             TestSuite suite = new VariosTestSuite("Pruebas varias Gestoli");
             suite.addTestSuite(UsuariTest.class);
             suite.addTestSuite(TipusEnvasTest.class);
             suite.addTestSuite(EstablimentTest.class);
             suite.addTestSuite(OlivicultorTest.class);
             return suite;
      }

}
```

Los procedimientos setUp() y tearDown() se ejecutan antes y después de cada uno de los otros métodos. El código de los otros procedimientos viene en gran parte de la exportación a jUnit de Selenium IDE después de haber registrado un script.

Para la ejecución de los tests:

- Con Selenium Server: se ejecuta con *java -jar selenium-server.jar*
- Con Eclipse: hay varias formas de ejecutar los tests desde Eclipse:
    - Haciendo click derecho en el nombre del método dentro de la Case y "run as... jUnit test" nos permite ejecutar solo la prueba de ese método
    - Haciendo click derecho en el nombre del archivo TestCase y "run as... jUnit test" nos permite ejecutar todas las pruebas que estén en esa Case secuencialmente.
    - Haciendo click derecho en el nombre del archivo TestSuite y "run as... jUnit test" nos permite ejecutar las pruebas de toda la Suite.

at4.net internet y comunicación . Santiago Rusiñol, 8. primero – Palma de Mallorca
www.at4.net tel. +34 971 22 88 70 – fax. +34 971 22 88 71 – info@at4.net

Selenium IDE a veces no pone una espera después de un assert de dialogo de confirmación que redirige a otra página. Si no se pone una espera puede haber resultados incorrectos, hay que ponerla manualmente.

Otras páginas de referencia:

- http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=seleniumIDE
- http://soyunjuncohueco.wordpress.com/2008/02/01/tests-de-aceptacion-automatizados-con-selenium/
- http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=selenium_rc
- http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=selenium_core

5.2 Tests "tradicionales"

Durante toda la fase de desarrollo de cada una de las funcionalidades de la aplicación, un miembro del equipo de desarrollo (distinto al que implementó la funcionalidad) realizó los tests unitarios y funcionales (siguiendo las directrices del documento de diseño proporcionado por IBIT) directamente sobre nuestro entorno de desarrollo.

Después de cada una de las distintas fases del proyecto, se realizó un exhaustivo test que consistía en vaciar totalmente la base de datos y un equipo de 3 o 4 personas realizaban tests de módulo y funcional de todas las funcionalidades. Estos tests se realizaron los días 1 y 2 de junio, 17 y 18 de junio, y del 28 al 31 de julio.

También, después de cada nueva implantación en el entorno de producción, se realizaron tests sobre este entorno.

6 IMPLANTACIÓN

La aplicación desarrollada en **JAVA JDK1.4** corre sobre un servidor de aplicaciones **JBoss 3.2.8-caib7** en un servidor **Linux**. La base de datos es **PostgreSQL 8.2.7**.

Para la instalación del lector RFID debe seguirse el documento "**GEST-OLI - Instalación lector rfid**", también proporcionado por **at4**.

at4.net internet y comunicación . Santiago Rusiñol, 8. primero – Palma de Mallorca
www.at4.net tel. +34 971 22 88 70 – fax. +34 971 22 88 71 – info@at4.net

En el servidor Linux es necesario que se instalen las librerías **XWindow**[12] (con el básico es suficiente), las cuales permiten utilizar un ambiente gráfico bajo Linux. Esto es necesario para la generación de los PDFs. También es importante que dentro de los parámetros java con que se arranca el JBoss se añada:

```
-Djava.awt.headless=true
```

Dentro del JBoss, en el archivo **conf/log4j.xml** se debe incluir:

```
<logger name="es.caib.gestoli">
   <level value="INFO"/>
</logger>
```

para fijar el nivel de log de las propias clases de la aplicación.

Para crear la base de datos:

```
CREATE DATABASE gestoli_db;
```

y posteriormente, utilizando el usuario de BBDD con el que se conectará la aplicación (se encuentra en el archivo *gestoli-ds.xml*) ejecutar los scripts de los archivos **scriptBD.sql** y **updateBD.sql**.

El formato de las fechas en la BBDD debe ser DMY, con lo cual se tiene que ejectuar:

```
ALTER DATABASE gestoli_db SET DateStyle = 'ISO, DMY';
```

y posteriormente comprobar que esté correcto:

```
gestoli_db=#  SHOW datestyle;
 DateStyle
-----------
 ISO, DMY
(1 fila)
```

El locale del servidor debe ser “es_ES”:

```
jboss@dgtic::~$ locale
LANG=es_ES.UTF-8
LC_CTYPE="es_ES.UTF-8"
LC_NUMERIC="es_ES.UTF-8"
LC_TIME="es_ES.UTF-8"
LC_COLLATE="es_ES.UTF-8"
LC_MONETARY="es_ES.UTF-8"
LC_MESSAGES="es_ES.UTF-8"
LC_PAPER="es_ES.UTF-8"
LC_NAME="es_ES.UTF-8"
LC_ADDRESS="es_ES.UTF-8"
LC_TELEPHONE="es_ES.UTF-8"
LC_MEASUREMENT="es_ES.UTF-8"
LC_IDENTIFICATION="es_ES.UTF-8"
```

12 http://en.wikipedia.org/wiki/X_Window_System

at4.net internet y comunicación . Santiago Rusiñol, 8. primero – Palma de Mallorca
www.at4.net tel. +34 971 22 88 70 – fax. +34 971 22 88 71 – info@at4.net

```
LC_ALL=
```

El EAR resultante de la aplicación (gestoli.ear) se encontrará en la carpeta *output/product*, del proyecto Eclipse, una vez se haya ejecutado el comando *ant make*. Este EAR deberá moverse a la carpeta *deploy* del JBoss. En esta misma carpeta debe copiarse el archivo gestoli-ds.xml y el contenido del mail-service.xml (ya explicados en un apartado previo de este documento).

Arrancamos el JBoss y hay que seguir ciertos pasos en el siguiente orden:

1. Ejecutar http://<dominio>/gestoli/InicialitzarBBDD.html. Esta acción inserta datos iniciales en varias tablas de la base de datos: oli_color, oli_arxiu, oli_material_tipus_envas, oli_material_diposit y oli_varietat_oliva. Al ejecutar la URL ya nos mostrá información de si el proceso ha sido correcto o no. De todas formas, revisar si las tablas mencionadas han sido informadas con datos, si alguna de estas está vacía, es por algún problema que se mostrará en la respuesta de la URL y en los logs. Para poder volver a ejecutar esta URL, previamente se tiene que borrar todo el contenido de estas tablas.
2. Dar de alta una campaña
3. Asignar nuevas tarifas

at4.net internet y comunicación . Santiago Rusiñol, 8. primero – Palma de Mallorca
www.at4.net tel. +34 971 22 88 70 – fax. +34 971 22 88 71 – info@at4.net