Universidad Autónoma Metropolitana
Unidad Azcapotzalco

División de Ciencias Básicas e Ingeniería

Proyecto Terminal en Ingeniería en Computación

Prototipo Web para Gestionar Proyectos de Software

Proyecto que presenta:
Ramirez Comonfort Paulino

para obtener el titulo de:
Ingeniero en Computación

Director de Proyecto:
M. en C. Oscar Alvarado Nava

México, D.F. **Enero de 2011**

Agradecimientos

A todas aquella personas que me apoyaron de diversas formas durante el transcurso de mi carrera y de este proyecto les agradezco sinceramente. Un agredecimiento especial para Belén, Alejandrina, Erika, Elvin, Irais y Eder que siempre me ofrecieron su apoyo total. También a esta casa de estudios, la UAM Azcapotzalco, por brindarme todas las herramientas necesarias para mi crecimiento académico, y por supuesto a todos mis maestros de la UAM, que fueron los medios por los que adquirí mis conocimientos, en especial para el M. Oscar.

Dedicatoria

Dedicada con todo mi amor y cariño a mis padres y hermanos.

Gracias a ti padre y madre por darme esta maravillosa oportunidad. A ti padre por todos los sacrificios que tuviste que hacer para poder darme esta educación y por inculcarme buenos principios. A ti madre porque sé que hiciste sacrificios para apoyarme en mis estudios. Papá y mamá gracias por ser pacientes para conmigo, y gracias por la gran confianza que me tienen. Y a mis hermanos Erika, Elvin, Irais y Eder porque me ofrecieron su apoyo total. Sé que en muchas ocasiones les hice falta por no estar físicamente con ustedes y les agradezco por comprenderme, por eso dedico este trabajo con mucho amor para ustedes. También a todos mis amigos que siempre me ofrecieron palabras de ánimo para terminar esta carrera que comencé años atrás. Sinceramente a todos ustedes, gracias.

Índice general

Índice de figuras

Índice de tablas

Introducción

0.1. Objetivo del documento

El presente documento es un reporte del trabajo realizado para desarrollar el *Prototipo Web para Gestionar Proyectos de Software*, es cual se describe detalladamente en [1].

0.2. Objetivos del Proyecto

El objetivo principal del proyecto fue:

- Implementar un prototipo de una aplicación web para gestionar proyectos de software que se desarrollen bajo el proceso RUP.

Los objetivos particulares fueron:

- Definir los requerimientos del sistema
- Elaborar la arquitectura del sistema
- Programar y probar los módulos del sistema
- Integrar los módulos
- Probar el sistema
- Implantar el sistema

En esta documentación del proyecto se describen la actividades desarrolladas que permitieron finalmente obtener un producto software acorde al título del proyecto.

Capítulo 1

Qué es el Proceso Unificado

1.1. Proceso de Desarrollo de Software

Debido a que el prototipo web que se pretende desarrollar con este proyecto estará adaptado al Proceso Unificado, comenzaremos por explicar lo que es un proceso de desarrollo de software y de cómo nace y funciona el Proceso Unificado.

Un proceso de desarrollo de software es el conjunto de actividades necesarias para transformar los requisitos de un usuario en un sistema software. Generalmente el objetivo de seguir un proceso de software es producir un producto software de forma eficaz y eficiente, y que además satisfaga las necesidades del cliente. A continuación describimos uno de los diferentes procesos de software que existe, el Proceso Unificado, conocido como RUP.

1.2. Historia del Proceso Unificado

El Proceso Unificado tiene sus raíces en los logros de Ericsson. Ericsson modelaba sus sistemas como un conjunto de bloques interconectados, es decir, utilizaba lo que hoy conocemos como *desarrollo basado en componentes*, cuyo creador fue Ivar Jacobson.

En 1987 Ivar Jacobson dejó Ericsson y fundó Objectory AB en Estocolmo. Durante los siguientes ocho años, él y sus colaboradores desarrollaron un proceso denominado Objectory, una abreviación de "Object Factory". Es en este, en donde se introduce el concepto *caso de uso*. El desarrollo del Proceso Objectory continuó en una serie de

versiones, desde Objectory 1.0 en 1988 a la primera versión iterativa, Objectory 3.8, en 1995.

A finales de 1995, Rational Software Corporation, compró Objectory AB dándose a la tarea de unificar los principios básicos subyacentes en los procesos de desarrollo existentes. Rational había desarrolado algunas prácticas de desarrollo de software, la mayoria de ellas complementarias a las contenidas en Objectory. Se dió énfasis a un proceso basado en la arquitectura y en el desarrollo iterativo. Una vez añadida la experiencia y prácticas de Rational surge el Proceso Objectory de Rational 4.1, el cual incluía las fases y las iteraciones. También se incorporó el lenguaje UML, que para ese entonces estaba en fase de desarrollo.

A mediados de 1998 el Proceso Objectory de Rational se había convertido en un proceso hecho y derecho, capaz de soportar el ciclo de vida del desarrollo en su totalidad. Para ello, integraba una amplia variedad de aportaciones, entre ellos de los autores de UML, las diversas empresas que Rational compró y de las que se fusionaron. En Junio, Rational publicó una nueva versión del producto, el Proceso Unificado de Rational 5.0. En ese momento, por primera vez, se pusieron a disposición del público en general muchos elementos de ese proceso propietario.

El cambio de nombre refleja el hecho de que la unificación ha tenido lugar en muchas dimensiones: unificación de técnicas de desarrollo, a través del Lenguaje Unificado de Modelado, y unificación del trabajo de muchos metodologistas[2].

1.3. Qué es RUP

El Proceso Unificado sirve como un proceso de desarrollo de software. Sin embargo, es más que eso, es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas software.

El Proceso Unificado está basado en componentes, es decir, el software en contrucción está formado por componentes software interconectados por medio de interfaces.

El Proceso Unificado utliza el Lenguaje Unificado de Modelado (Unified Modeling Language, UML) para modelar todos los esquemas de un sistema software.

Las características claves del Proceso Unificado son: es dirigido por casos de uso, centrado en la arquitectura, iterativo e incremental.

El Proceso Unificado está dirigido por Casos de Uso. Un caso de uso es una funcionalidad del sistema que proporciona al usuario un resultado importante. La importancia de los casos de uso radica en que, además de especificar los requisitos de un sistema, también guían su diseño, implementación y prueba, en otras palabras, guían el proceso de desarrollo. En el caso del Proceso Unificado, en el proceso de

desarrollo se avanza a través de una serie de flujos de trabajo que parten de los casos de uso.

El Proceso Unificado está centrado en la arquitectura. La arquitectura en un sistema software se describe mediante diferentes vistas del sistema en contrucción. La descripción de la arquitectura tiene cinco secciones, una para cada modelo. Tiene una vista del modelo de casos de uso, una vista del modelo de análisis, una vista del modelo de diseño, una vista del modelo de despliegue, y una vista del modelo de implementación.

Vista del modelo de casos de uso. Presenta los actores y casos de uso más importantes.

Vista del modelo de diseño. Presenta los clasificadores más importantes para la arquitectura pertenecientes al modelo de diseño: los subsistemas, las interfaces más importantes, así como algunas pocas clases muy importantes. También presenta las realizaciones de casos de uso.

Vista del modelo de despliegue: Define la arquitectura física del sistema por medio de nodos interconectados. Cada nodo es un elemento hardware sobre los cuales se pueden ejecutarse los elementos software.

Vista del modelo de implementación. Es una correspondencia directa de los modelos de diseño y de despliegue.

El Proceso Unificado es iterativo e incremental. En proyectos muy grandes es práctico dividir el trabajo en partes más pequeñas o miniproyectos. Cada miniproyecto es una iteración que resulta en un incremento. Los desarrolladores basan la selección de lo que se implementará en una iteración en dos factores. En primer lugar, la iteración trata un grupo de casos de uso que juntos amplían la utilidad del producto desarrollado hasta ahora. En segundo lugar, la iteración trata los riesgos más importantes. Al ser miniproyectos, la iteración comienza con los casos de uso pasando al análisis, diseño, implementación y prueba.

1.3.1. La vida del Proceso Unificado

El Proceso Unificado se repite a los largo de una serie de ciclos que constituyen la vida de un sistema, como se observa en la figura 1.1 [1]. Cada ciclo concluye con una versión del producto para los clientes. Cada ciclo consta de cuatro fases: inicio, elaboración, construcción y transición. Cada fase se subdivide a su vez en iteraciones.

Fase de Inicio. Durante la fase de inicio se desarrolla una descripción del producto final a partir de una buena idea y se presenta el análisis de negocio para el producto.

[1] Fuente: http://phylum.com.mx/

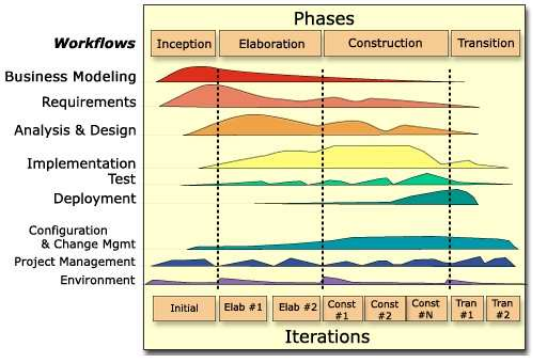

Figura 1.1: Ciclo de vida del proceso RUP.

Básicamente se responde a las preguntas:

- ¿Cuáles son las principales funciones del sistema para sus usuarios más importantes?
- ¿Cómo podría ser la arquitectura del sistema?
- ¿Cuál es el plan de proyecto y cuánto costará desarrollar el producto?

Fase de Elaboración. Durante la fase de elaboración, se especifican en detalle la mayoría de los casos de uso del producto y se diseña la arquitectura del sistema. La arquitectura se expresa en forma de vistas de todos los modelos del sistema, los cuales juntos representan al sistema entero. Durante esta fase se implementan los casos de uso más críticos del sistema. Al final de la fase, el director del proyecto está en disposición de planificar las actividades y estimar los recursos para terminar el proyecto.

Fase de Construcción. En esta fase se crea el producto, el software ya terminado. Al final de esta fase, el producto contiene todos los casos de uso que la dirección y el cliente acordaron.

Fase de Transición. Cubre el periodo durante el cual el producto se convierte en versión beta. En la versión beta un número reducido de usuarios con experiencia prueba el producto e informa de los defectos y deficiencias. Los desarrolladores corrigen los problemas e incorporan algunas de las mejoras sugeridas. La fase de transición conlleva actividades como la fabricación, formación del cliente, el proporcionar una linea de ayuda y asistencia, y la corrección de defectos que se encuentren tras la entrega.

1.4. Cómo funciona RUP

RUP como se mencionó anteriormente, tiene como característica la división de un proyecto en miniproyectos, en donde, cada miniproyecto tiene objetivos bien específicos que permiten ir contruyendo el sistema hasta completarse. A continuación explicaremos de forma breve en que consiste la iteración.

1.5. Iteración

Una iteración es un miniproyecto,– un recorrido más o menos completo de todos los flujos de trabajo fundamentales (requisitos, análisis, diseño, implementación y prueba)–, que obtiene como resultado una versión iterna. Cada iteración implementa ciertos casos de uso, elegidos previamente. El objetivo de dividir cada fase en iteraciones es: planificar un poco, especificar, diseñar e implementar un poco e integrar, probar y ejecutar un poco en cada iteración.

Cada iteración consta de los flujos principales del proyecto general: planificación, desarrollo y una preparación para la entrega. Al final de cada iteración hay un resultado tangible que formará parte del entregable final o del sistema. Esto es útil ya que permite decidir si continuar o no con iteraciones o fases subsiguientes, además de que se obtiene retroalimentación después de cada iteración. También las iteraciones ayudan a planificar, organizar y tener un seguimiento correcto del proyecto.

Capítulo 2

Tecnologías Utilizadas

Durante el desarrollo de este proyecto se utilizaron diversas tecnologías para implementar las funcionalidades que requirió el sistema. En las secciones siguientes daremos una descripción breve de todas ellas.

2.1. Struts 2

Struts es un framework desarrollado y mantenido por la *Apache Software Foundation*. Es un framework para el desarrollo de aplicaciones Web basadas en la tecnología Java. Struts 2 es un tanto diferente de Struts 1, ya que esta versión se basa más en un framework de desarrollo Java llamado WebWork, el cual se considera superior a Struts 1.

Struts 2 está basado en el patrón MVC(Modelo-Vista-Controlador), una arquitectura que busca reducir el acoplamiento separando la aplicación en tres capas. El MVC es una forma de organizar el código de la aplicación separando los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes, que describimos a continuación.

El modelo: Tiene la responsabilidad de acceder a los datos. Define también las reglas del negocio. Por tanto, hace referencia a los datos que maneja la aplicación y las reglas de negocio que operan sobre ellos y que se traducen en Struts 2 a las acciones.[1]

[1]Entiéndase acción como una operación que puede realizar el sistema, tales como eliminar, borrar, etc.

La vista: Se encarga de generar la interfaz con la que la aplicación interacciona con el usuario. En Struts 2 equivale a los resultados.

El controlador: Recibe los eventos de entrada y los gestiona. Comunica la vista y el modelo respondiendo a eventos generados por el usuario en la vista, invocando cambios en el modelo, y devolviendo a la vista la información del modelo necesaria para generar la respuesta adecuada para el usuario. El controlador se implementa en Struts 2 mediante el filtro FilterDispatcher.[2]

La ventaja de utilizar estos componentes, es decir, trabajar con MVC, radica en la facilidad para realizar cambios en la aplicación. Cuando se realiza un cambio en base de datos, programación o interfaz de usuario sólo tocaremos uno de los componentes. Además se pueden modificar sólo uno de los componentes sin conocer cómo funcionan los demás.

Sobre esta tecnología se desarrolla todo el sistema del presente proyecto.

2.2. Hibernate

Hibernate es una herramienta de tipo ORM(Object-Relational-Mapping). Un ORM es un Mapeo de Objetos BD (Base de Datos) Relacionales, consisten en una serie de objetos que permiten acceder a los datos y que contienen en su interior cierta lógica de negocio. El objetivo es utilizar Objetos en vez de registros y Clases en vez de tablas. Ya que las bases de datos tienen una estructura relacional, para acceder a la BD como si fuera orientada a objetos se necesita de una interfaz que traduzca la lógica de objetos a la lógica relacional, esta interfaz es lo que se denomina ORM, el cual es una capa de abstracción.

Hibernate, nos permite almacenar objetos de aplicaciones Java en tablas de sistemas de bases de datos relacionales usando metadatos que describen la relación entre los objetos y la base de datos, haciendolo de forma transparente.

Para usar Hibernate hay varias formas de hacerlo. Para nuestro caso, seguiremos los siguientes pasos:

1. Crear la BD
2. Integrar Hibernate con Struts
3. Crear las clases a ser mapeadas
4. Mapear las clases

[2]Clase perteneciente al framework Struts 2 que gestiona las acciones invocadas por el usuario.

Estos pasos los iremos describiendo, aunque con respecto a la configuración de Hibernate para su integración con Struts hay gran cantidad de documentación por la red y en la propia página de Hibernate, por lo que no repetiremos dicha información. Por otra parte la creación de las clases y el mapeo se expondrán en su momento. Cabe mencionar que para realizar el mapeo de las clases a la base de datos, hay dos formas de hacerlo: archivos xml o usar anotaciones.

2.2.1. Mapeo de Clases con archivos XML

Se utilizan archivos xml para proporcionar la información que permitirá a Hibernate saber la correspondencia entre nuestra clase y la tabla en la bases de datos. Por ejemplo, supongamos que tenemos una clase llamada Rol, de la siguiente forma:

```
1 public class Rol implements Serializable{
  private String nombreRol;
3 private String descripcion;
  private String urlDescripcion;
5 private String categoria;

7 /**
    * Metodos getter y setter
9   */
}
```

Listing 2.1: Ejemplo de clase Rol.java

Supongamos que queremos mapear esta clase a la tabla ROL, como la mostrada en la figura 2.1.

Figura 2.1: Tabla Rol

EL formato del archivo sería el siguiente:

```
<?xml version=``1.0" encoding=``UTF-8"?>
2 <!DOCTYPE hibernate-mapping PUBLIC
  ``-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4 ``http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping>
```

```
6   <class name=``Rol" table=``Rol">
      <id name=``id" column=``nombreRol">
8       <generator class=``assigned"></generator>
      </id>
10    <property name=``descripcion" column=``descripcion"/>
      <property name=``urlDescripcion" column=``urlDescripcion"></property
          >
12    <property name=``categoria" column=``categoria"></property>
    </class>
14 </hibernate-mapping>
```

Listing 2.2: Ejemplo de mapeo de la clase Rol.java a la tabla Rol

Como habrá apreciado, tiene que haber una correspondencia entre los tipos de datos de la clase con la tabla. Además vemos que el archivo lleva una sintaxis. La palabra clave *class* se usa para indicar el nombre de la clase, mientras que la palabra *TABLE* para indicar el nombre de la tabla a la que se mapeará. Con la palabra *id* especificamos cuál es el atributo que va asociado a la columna que es llave primaria en la tabla, así como la forma en que es generada la llave primaria, para este caso utilizamos la palabra *class="assigned"*, que le indica a Hibernate que será la aplicación quien asigne este valor. Posteriormete utilizando la palabra *property* indicamos el nombre del atributo de la clase y la columna a la cual se mapeará. Por ejemplo, la propiedad de la clase *descripcion* va mapeada a la columna *descripcion* de la tabla, y así por el estilo con las otras propiedades de la clase.

Lo anterior evidentemente es un ejemplo muy sencillo de cómo se hace el mapeado en Hibernate. En una aplicación de tamaño mediano las relaciones entre clases tendrán que mapearse, tales como 1:1, 1:N o N:N, las cuales explicaremos en su momento.

2.2.2. Mapeo de Clases con Annotations

En programación Java, una Anotación Java es una forma de agregar metadatos al código que estará disponible para la aplicación en tiempo de ejecución. Las anotaciones tienen diferentes usos: Provee información para el compilador, proveer información que se utilizará para la generación del código o información a ser utilizada por la aplicación en tiempo de ejecución. A continuación mostramos algunos ejemplos de anotaciones Java.

```
@Author(
2   name = "Benjamin Franklin",
    date = "3/27/2003"
4 )
class MyClass() { }
```

Listing 2.3: Ejemplo 1 de anotaciones

Otro ejemplo:

```
@Override
void mySuperMethod() { }
```

Listing 2.4: Ejemplo 2 de anotaciones

Visto entonces que las anotaciones tienen diversos usos en la programación, queremos mencionar que el mapeado de una clase Java a una tabla en la BD se puede hacer usando anotaciones Java. Para esto, Hibernate provee en su página documentación suficiente sobre el uso de estas anotaciones. Regresando a nuestro ejemplo anterior, a continuación mostramos cómo hariámos el mapeado de la clase Rol a la tabla Rol.

```
@Entity
public class Rol implements Serializable{
  private String nombreRol;
  private String descripcion;
  private String urlDescripcion;
  private String categoria;

  public Rol(){

  }
  @Id
  public String getNombreRol() {
    return nombreRol;
  }
  public void setNombreRol(String nombreRol) {
    this.nombreRol = nombreRol;
  }
  @Column(name=``descripcion")
  public String getDescripcion() {
    return descripcion;
  }
  public void setDescripcion(String descripcion) {
    this.descripcion = descripcion;
  }
  public String getUrlDescripcion() {
    return urlDescripcion;
  }
  public void setUrlDescripcion(String urlDescripcion) {
    this.urlDescripcion = urlDescripcion;
  }
  public String getCategoria() {
    return categoria;
  }
```

Listing 2.5: Mapeado usando Anotaciones

Al utilizar la palabra *@Entity* estamos indicando que esta clase será mapeada a una tabla. Al utilizar anotaciones, si no indicamos el nombre de la tabla, Hibernate

asume que la tabla tiene el mismo nombre que el de la clase, en este caso "Rol", por eso no hemos indicado el nombre. Posteriormente utilizamos la palabra *@Id* para indicar el atributo que será mapeado a la columna que sirve de llave primaria. Para los otros atributos usamos la palabra *@Column* para especificar el nombre de la columna a la que irán mapeadas. Auque aquí sólo hemos usado *@Column* para el atributo *descripcion* debido que el nombre de las otras propiedades es el mismo en la tabla, y como se explicará posteriormente, en esos casos no es necesario indicarle a hibernate la columna.

Estos son ejemplos muy sencillos de cómo funciona el mapeado en Hibernate. Una tecnologiá que simplifica el acceso a los datos desde nuestra aplicación Java.

2.3. MySQL

La Base de Datos es una serie de datos organizados y relacionados entre si, los cuales son recolectados y explotados por los sistemas de una empresa o negocio en particular. Existe un interfaz entre el usuario y las aplicaciones que utilizan la base de datos, esa interfaz se conoce con el nombre de Sistema de Gestión de Datos (SGBD). Se compone de un lenguaje de definición de datos, un lenguaje de manipulación de datos y un lenguaje de consulta.

MySQL es un SGBD para bases de datos relacionales, multihilo y multiusuario. MySQL fue escrito en C y C++ pero se adapta a diferentes entornos de desarrollo. Por ejemplo es capaz de interactuar con PHP, Perl y Java, así como integrarse a diferentes sistemas operativos. Actualmente MySQL es el SGBD open source más popular, su continuo desarrollo y creciente popularidad está haciendo de MySQL el competido directo de Oracle.[3].

En este proyecto usaremos la versión 5.1.41 para el sistema operativo Linux.

2.4. CSS

Las CSS (Cascading Style Sheets), es una tecnología desarrollada por la World Wide Web Consortium (W3C)[4] con el fin de controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML, el cual es muy útil al momento de crear páginas web complejas.

Al crear una página web, se utiliza el lenguaje HTML/XHTML para marcar los

[3]http://www.esepestudio.com/articulo/desarrollo-web/bases-de-datos-mysql/Que-es-MySQL.htm, Junio 2010

[4]http://www.w3.org/

contenidos, es decir, especificar la función de cada elemento en la página: un título, un párrafo, una lista, una tabla, etc. Una vez creados los contenidos, se utilizan los CSS para definir el cómo se mostrarán los elementos, es decir, su presentación. De esta forma, con los CSS se define su color, tamaño, tipo de letra, posición de cada elemento, etc.

Si bien los CSS ofrecen una amplia posibibidad a la hora de diseñar la presentación, el trabajo del diseñador siempre estará limitada a las posibilidades del navegador. La razón es que cada navegador tiene una parte llamada motor, que se encarga de interpretar el HTML y CSS, por lo que es ese motor el que pondrá los límites en cuando a lo que puede interpretar y lo que no.

La especificación oficial que se utiliza para diseñar páginas web con CSS es la versión CSS 2.1.

2.4.1. Funcionamiento de CSS

Antes de que surgieran los CSS, el diseñador web tenía que definir el aspecto de cada elemento con etiquetas HTML, he aquí un ejemplo:

```
<h1><font color="red" face="Arial" size="5">Titular de la página</font></h1>
```

El problema de lo anterior, radica en que si se tienen 50 elementos diferentes, habría que insertar 50 etiquetas font, si el sitio web completo tuviera 10,000 páginas habría que definir 500,000 etiquetas font, lo cual no es muy agradable. Con las hojas de estilo, la situación es diferente. Como se mencionó anteriormente, primero se utiliza HTML/XHTML para marcar los contenidos. Retomando el ejemplo, nos quedaría como sigue:

```
<h1>Titular de la página</h1>
```

Con lo anterior sólo estamos diciendo que este texto es un título. Una vez realizado lo anterior, ahora nos preocuparemos de la presentación. Esto se logra utilizando hojas CSS, que se definen en un archivo por separado o bien en el *head* de la página, tomando una forma como la siguiente:

```
h1 { color: red; font-family: Arial;font-size: 10px; }
```

Con lo anterior estamos especificando que todas los títulos marcados con la etiqueta *h1* deberán de mostrarse en color rojo, con el tipo de letra Arial y con tamaño

de letra 10 pixeles. La ventaja de esto es que si en algún momento se desea cambiar la presentación de todos los titulos marcados con *h1* sólo tenemos que modificar el archivo de la hoja de estilo, sin tocar el contenido.

Esto es en forma suscinta el funcionamiento de la tecnología CSS.

2.5. Java

Java es un lenguaje de programación creada por Sun Microsystems en 1995. Algunas de las principales carácteristicas de Java son:

- Lenguaje totalmente orientado a objetos.
- Disponilidad de un amplio conjunto de librerías.
- Aplicaciones multiplataforma.
- Ejecución segura de aplicaciones.
- Amplio soporte de fabricantes de software.

Una de las caráteristicas importantes de Java es la Máquina Virtual de Java (JVM). La JVM es un entorno de ejecución para aplicaciones Java que se encarga de adaptar los programas Java compilados a las caracteristicas del sistema operativo objetivo. Todo programa Java se organiza en clases, estas a su vez se codifican en archivos de texto con extensión .java. Estos archivos de código fuente se compilan generando un archivo .class que son los bytecodes, independientes de la arquitectura. Esto implica que los bytecodes no pueden ser ejecutados directamente por ningún sistema operativo, es durante la fase de ejecución cuando los bytecodes se someten a un proceso de interprestación, es decir, se traducen los bytecodes a código ejecutable por el sistema operativo. Esta operación es realizada precisamente por la JVM. En la figura 2.2 se muestra gráficamente el proceso descrito.

Cada sistema operativo provee su propia implementación de la JVM. Hoy en día podemos encontrar la JVM para la mayoría de los sistemas operativos. Es esta característica de Java, aunada a las ya mencionadas, las que han permitido que sea actualmente el lenguaje de programación más utilizado por la comunidad de desarrolladores.

Actualmente Java tiene diversas ediciones, que son:

- Java 2 Standar Edition (J2SE). Conformado por clases de uso general.

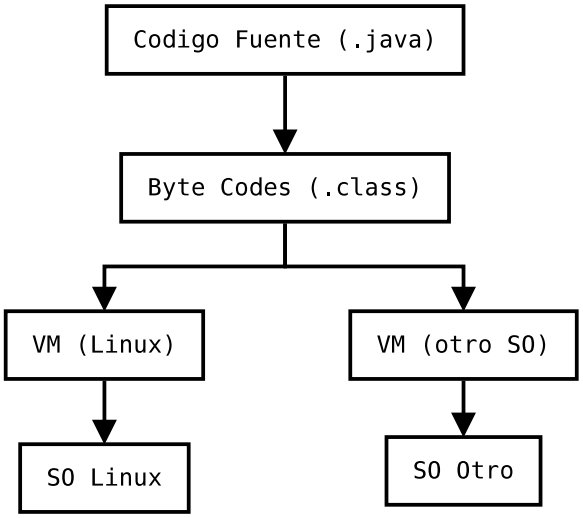

Figura 2.2: Proceso de Compilación Java

- Java 2 Enterprise Edition (J2EE). Proporciona los paquetes y tecnologías necesarias para la creación de aplicaciones empresariales multicapa.
- Java 2 Micro Edition (J2ME). Proporciona los paquetes y especificaciones para la creación de aplicaciones para dispositivos electrónicos de capacidades limitadas.

En este proyecto se usará la edicion J2EE.

2.6. JSP

Java Server Pages (JSP) es una tecnología que permite crear contenidos dinámicos del lado del servidor. Tiene un sintaxis parecida a Java.[5]

2.7. JavaScript

JavaScript es un lenguaje interpretado que permite incluir macros en páginas web. Estas se ejecutan del lado del cliente y no en el servidor. JavaScript nos permite:

- Controlar ventanas y contenidos a mostrar.
- Programar páginas dinámicas simples.

[5]http://java.sun.com/products/jsp/

- Capturar eventos generados por el usuario y responder a ellos procesando en el cliente.
- Validación de datos antes de enviarlos al servidor.
- Comunicación/Interacción con el usuario.

2.8. JQuery

JQuery es una biblioteca de JavaScript creada con el fin de simplificar la manera de interactuar con los documentos HTML, manipular el arbol DOM (Document Object Model), manejar eventos, desarrollar animaciones y agregar interacción con la tecnología Ajax. JQuery es software libre y de código abierto. Ofrece una serie de funcionalidades basadas en JavaScript que de otra forma requerirían mucho más código.[6]

2.9. JSON

JavaScript Object Notation (JSON) es un formato ligero para el intercambio de datos. Es un formato alternativo de envío y recepción de información, reemplaza a XML o el envío de texto plano. Hace el código más sencillo ya que utiliza el código JavaScript como modelo de datos. Por ejemplo para definir una estructura de datos usando XML, un ejemplo nos quedaría de la siguiente forma:

```
<persona>
   <nombre>juan</nombre>
   <edad>22</edad>
   <estudios>
      <estudio>primario</estudio>
      <estudio>secundario</estudio>
   </estudios>
</persona>
```

Por otra parte, la definición de una estructura de datos utilizando JSON es muy similar a como lo haríamos con JavaScript. Utilizando JavaScript la definción de la estructura se haría de la siguiente forma:

[6] http://jquery.com/

```
var persona={
             'nombre':'juan',
             'edad':22,
             'estudios':['primario','secundario']
            };
```

Ahora, si utilizamos JSON, sería asi:

```
{
  'nombre':'juan',
  'edad':22,
  'estudios':['primario','secundario']
}
```

Como se aprecia, con JSON se declaran directamente las propiedades y su valor, separadaos por coma. Los anterior evidentemente resultará mucho más ligero que si utilizáramos XML. Además, al recuperar la información en una estructura JSON es muy fácil, a diferencia de la recuperación en un archivo XML.

2.10. Ajax

Ajax, un término presentado por primera ocasión en el artículo *"Ajax: A New Approach to Web Applications"*, publicado en 2005 por Jesse James Garrett, es un acrónimo de Asynchronous JavaScript + XML. Dicho artículo definía Ajax de la siguiente forma:

"Ajax no es una tecnología en sí mismo. En realidad, se trata de varias tecnologías independientes que se unen de formas nuevas y sorprendentes."

Dichas tecnologías son:

- XHTML y CSS, para crear una presentación basada en estándares.
- DOM, para la interacción y manipulación dinámica de la presentación.
- XML, XSLT y JSON, para el intercambio y la manipulación de información.
- XMLHttpRequest, para el intercambio asíncrono de información.
- JavaScript, para unir todas las demás tecnologías.

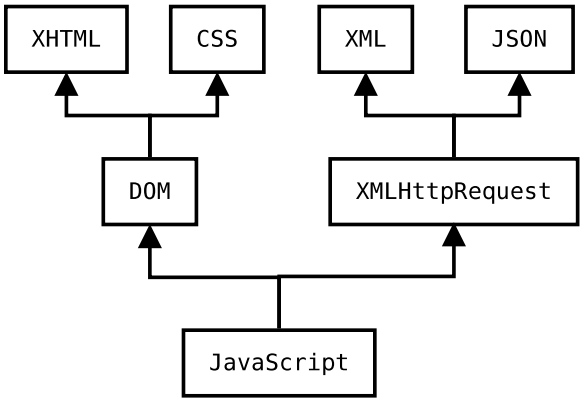

Figura 2.3: Tecnologías que engloba Ajax

En la figurar 2.3 mostramos un diagrama de las tecnologías mencionadas:

En las aplicaciones web tradicionales, cada acción del usuario desencadena una petición al servidor, se envía la petición al servidor, se procesa y se obtiene una página HTML en el navegador del usuario. El problema de esto, es que cuando se generan muchas peticiones al servidor, el usuario tiene que esperar la recarga de toda la página. Las aplicaciones construidas con Ajax eliminan esa recarga constante de páginas mediante la creación de un elemento intermedio entre el usuario y el servidor. Se mantiene una comunicación asíncrona con el servidor, lo cual permite realizar pequeños cambios sobre la página y no recargarla completamente, esto aumenta la interactividad, velocidad y usabilidad en la aplicación. La mayoriá de las peticiones Ajax, se efectuán mediante JavaScript.

En este proyecto sólo haremos uso de Ajax de forma muy limitada, sobre todo en aquellas operaciones en las que se necesite transferir gran cantidad de información utilizando JSON.

Capítulo 3

Requerimientos del Sistema

3.1. Requerimientos

Un requerimiento o un requisito se define como una capacidad que el sistema debe proporcionar.

Hay diferentes clases de requerimientos. El Proceso Unificado los clasifica de la siguiente forma.

- Funcionalidad
- Usabilidad
- Fiabilidad
- Rendimiento
- Compatibilidad

En este proyecto realizado, sólo nos centramos en los requerimientos funcionales y un poco en los requisitos de usabilidad. Los requerimientos de tipo funcional especifican acciones que el sistema debe ser capaz de realizar. Estos son con frecuencia descritos en los casos de uso. Los requerimientos de usabilidad tienen que ver con frecuencia con el factor humano (ejemplo, crear un diseño centrado en el usuario), la estética, consistencia en la interfaz de usuario, ayuda en linea, documentación para el usuario y material de capacitación. En las secciones siguientes describimos los flujos de trabajo realizados para conjuntar los requerimientos del sistema.

3.2. Análisis del Problema

Este flujo de trabajo tiene por objetivo llegar a comprender el problema que se va a resolver. Consiste en identificar a los involucrados, los límites y las restricciones del sistema a desarrollar. El Proceso Unificado establece cuatro actividades que hay que realizar para llegar a comprender el problema:

- Encontrar Actores y Casos de Uso
- Desarrollar la Visión
- Desarrollo de un Plan de Manejo de Requerimientos
- Captura del Vocabulario Común

A continuación se describirán sólo las actividades abordades en el proyecto.

3.2.1. Encontrar Actores y Casos de Uso

Una vez realizado el proceso de análisis del problema, se definieron diferentes casos de uso. A continuación se describen de forma detallada los casos de uso. Estos casos de uso se fueron refinando a lo largo del proyecto, los cuales se presentarán en 3.2.3.

3.2.2. Identificación de Actores

La metodología RUP define diversos roles que puede desempeñar cada participante del proyecto. A continuación se mencionan dichos roles.

- Analista
- Desarrollador
- Gestor
- Soporte y Produccion
- Tester
- Otro

RUP en realidad define roles más especializados para cada rol, por ejemplo para el rol Analista, se define un rol más específico como Analista del Proceso de Negocio, Analista del Sistema, entre otros. Aunque para cada tipo de Rol hay roles más especializados, para el sistema a desarrollar sólo divideremos en dos categorías a los usuarios del sistema. Por una parte tendremos a los usuarios que tengan rol de Analista, Desarrollador, Soporte y Producción, Tester y Otro que los identificaremos como el Actor Usuario. Por otra parte tendremos a los usuarios que tienen el rol de Gestor que lo identificaremos como el Actor Gestor de Proyecto. Por lo tanto para nuestro sistema sólo tendremos dos actores:

- Actor Gestor de Proyecto
- Actor Usuario

3.2.3. Diagramas de Casos de Uso

La figura 3.1 muestra el diagrama de casos de uso para el Gestor de Proyecto.

La figura 3.2 muestra el diagrama de casos de uso para el actor Usuario.

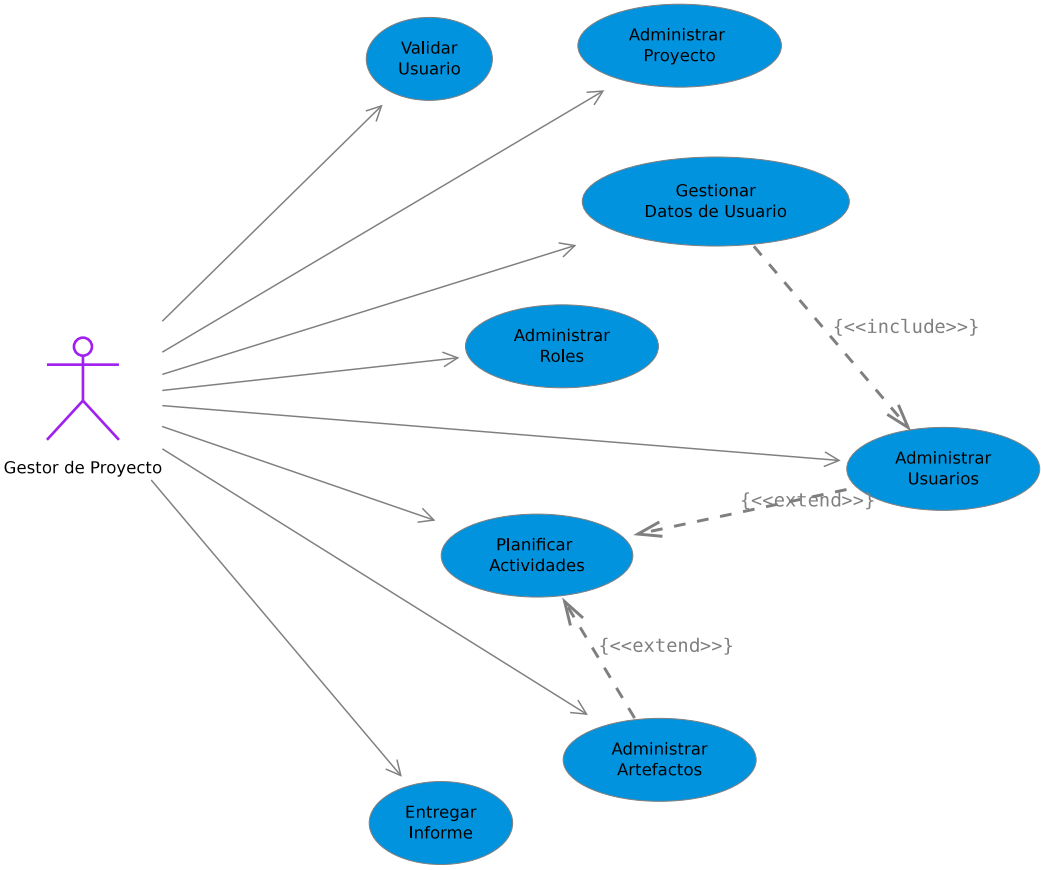

Figura 3.1: Diagrama de Casos de Uso para el Actor Gestor de Proyecto

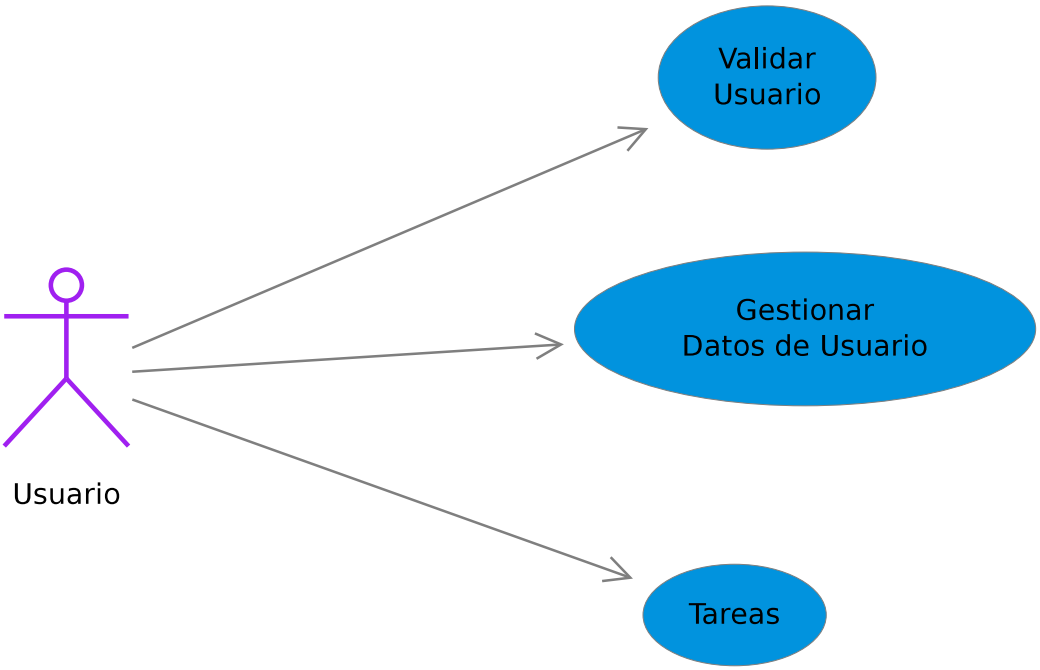

Figura 3.2: Diagrama de Casos de Uso para el Actor Usuario

3.2.4. Validar Usuario

Caso de Uso: Validar Usuario

Descripción En este caso de uso el sistema le permite al usuario autentificarse en el sistema.

Actores

Gestor de Proyecto, Usuario

Precondiciones

1. Haber iniciado la aplicación.

Flujo de Eventos

Flujo Básico

1. El sistema le presenta al usuario una pantalla en donde le solicita su login y su password.
2. El usuario introduce los datos solicitados.
3. El sistema valida los datos y le da acceso al usuario.

Flujos Alternativos

Excepciones

1. Si los datos del usuario no existen en el sistema, el sistema le indicara con un mensaje dicho error.

Puntos de extensión

1.

Postcondiciones

1. El usuario accede al sistema y puede realizar las tareas de los casos de uso.

3.2.5. Administrar Proyectos

Caso de Uso: Administrar Proyectos

Descripción Este caso de uso permite al gestor de proyectos realizar las operaciónes crear, modificar o eliminar un proyecto.

Actor Principal

Gestor de Proyectos

Precondiciones

1. El usuario debe haberse validado en el sistema y ser un Gestor de Proyecto.

Flujo de Eventos

Flujo Básico

1. El sistema le presenta al usuario la lista de proyectos del sistema a los que tiene permiso y un menú "Crear Proyecto", "Modificar Proyecto", "Eliminar Proyecto", "Gestionar Proyecto".
2. El usuario selecciona un proyecto.
3. El sistema muestra la información relacionada con ese proyecto.

Flujos Alternativos

Crear Proyecto

1. El sistema solicita los datos del nuevo proyecto.
2. El usuario introduce los datos necesarios.
3. El sistema registra el nuevo proyecto, creado su estructura general.
4. El sistema presenta la información registrada del nuevo proyecto.

Modificar Proyecto

1. El sistema le muestra al usuario la lista de proyectos sobre los que tiene permiso.
2. El usuario selecciona un proyecto.

3. El sistema muestra la información de dicho proyecto.
4. El usuario realiza los cambios pertinentes.
5. El sistema actualiza la información del proyecto en cuestión en el medio de almacenamiento.

Eliminar Proyecto

1. El sistema muestra al usuario la lista de proyectos sobre los que tiene permiso.
2. El usuario elige un proyecto.
3. El sistema muestra información de dicho proyecto.
4. El usuario solicita explícitamente borrar dicho proyecto.
5. El sistema pide confirmación.
6. El sistema elimina la información de dicho proyecto.

Gestionar Proyecto

1. El sistema solicita que el usuario elija un proyecto.
2. El usuario elige un proyecto.
3. El sistema presenta la interfaz necesaria para las actividades de gestión.

Excepciones

1.

Puntos de extensión

1.

Postcondiciones

1. La información es actualizada en el medio de almacenamiento.

3.2.6. Administrar Roles

Caso de Uso: Administrar Roles

Descripción En este caso de uso el usuario puede dar de alta un rol, modificar o eliminar uno existente.

Actor Principal

Gestor de Proyecto

Precondiciones

1. El usuario debe haberse validado en el sistema.

Flujo de Eventos

Flujo Básico

1. El sistema le muestra al usuario la lista de roles existentes en el sistema y un menú con las opciónes de Agregar, Modificar o Eliminar un Rol.
2. El usuario solicita al sistema agregar un nuevo rol.
3. El sistema presenta los campos que deben ser capturados para el crear el nuevo rol.
4. El usuario proporciona la información solicitada.
5. El sistema agrega el nuevo rol.
6. Se regresa al paso 3 si se desea agregar otro rol, de lo contrario se regresa al paso 1.

Flujos Alternativos

Modificar un Rol

1. El sistema muestra al usuario los datos del rol, con la posibilidad de modificarlos.
2. El usuario modifica los datos del rol.
3. El sistema guarda los cambios realizados y se regresa al paso 1 del Flujo Principal.

Eliminar un Rol

1. El usuario elige el rol a eliminar.
2. El sistema le muestra al usuario la información actual del rol.
3. El usuario solicita eliminar el rol.
4. El sistema pide confirmación.
5. El usuario confirma la eliminación.
6. El sistema elimina el rol y regresa al paso 1 del Flujo Principal.

Excepciones

1. En cualquiera de los flujos, si no se puede realizar la operación el sistema se lo indicará al usuario con un mensaje.

Puntos de extensión

1.

Postcondiciones

1. La información relacionada a roles queda en un estado consistente.

3.2.7. Administrar Usuarios

Caso de Uso: Administrar Usuarios

Descripción Este caso de uso permite al gestor de proyectos realizar las operaciónes crear, modificar o eliminar un usuario.

Actor Principal

Gestor de Proyecto

Precondiciones

1. El usuario debe haberse validado en el sistema y ser un Gestor de Proyecto.

Flujo de Eventos

Flujo Básico

1. El sistema le presenta al usuario la lista de usuarios del proyecto actual y un menú con las siguientes opciónes: “Modificar Usuario”, “Eliminar Usuario”, “Ver Datos del usuario” y adicionalmente una opción para “Registrar Usuario”.
2. El usuario elige un usuario.
3. El sistema comienza alguno de los flujos alternativos de acuerdo al botón que el usuario elija.

Flujos Alternativos

Ver Datos del Usuario

1. El sistema le muestra la usuario los datos del usuario elegido.

Registrar Usuario

1. El sistema solicita los datos del nuevo usuario.
2. El usuario proporciona los datos solicitados.
3. El sistema los valida y almacena en un medio de almacenamiento.

Modificar Usuario

1. El sistema le mostrará al usuario la lista de usuarios del proyecto actual.
2. El usuario selecciona un usuario.
3. El sistema muestra al usuario los datos con la posibilidad de modificarlos.
4. El usuario realiza los cambios pertinentes.
5. El sistema actualiza la información del usuario en cuestión en el medio de almacenamiento.

Eliminar Usuario

1. El sistema muestra al usuario la lista de usuarios del proyecto actual.
2. El usuario elige un usuario.

3. El sistema muestra la información de dicho usuario.
4. El usuario solicita explícitamente borrar dicho usuario.
5. El sistema elimina la información de dicho usuario del medio de almacenamiento.

Excepciones

1. En el flujo básico, si el usuario no ha seleccionado ninguno proyecto sobre el cual trabajar, la lista de usuarios se mostrará vacía.
2. En "Registrar Usuario", si los datos proporcionados son incorrectos el sistema le mostrará al usuario un mensaje de error, tras lo cual solicitará nuevamente que instroduzca los datos.
3. En "Modificar Usuario" si por alguna razón no se puede llevar a cabo correctamente la operación actualizar información, el sistema le mostrará un mensaje de error al usuario.

Puntos de extensión

1.

Postcondiciones

1. La información es actualizada en el medio de almacenamiento.

3.2.8. Gestionar Datos de Usuario

Caso de Uso: Gestionar Datos de Usuario

Descripción Permite al usuario modificar sus datos personales.

Actor Principal

Gestor de Proyecto, Usuario

Precondiciones

1. Haberse validado en el sistema.

Flujo de Eventos

Flujo Básico

1. El usuario solicita modificar sus datos personales.
2. El sistema le muestra al usuario sus datos personales con la posibilidad de modificarlos.
3. El usuario modifica los campos que cree convenientes y solicita al sistema que guarde los cambios.
4. El sistema graba los cambios realizados.

Flujos Alternativos
Cambio de Constraseña.

1. El sistema le solicita al usuario su contraseña actual.
2. El usuario introduce su contraseña actual.
3. El sistema verifica la contraseña y solicita la nueva contraseña dos veces.
4. El usuario introduce la nueva contraseña dos veces.
5. El sistema graba la nueva contraseña.

Excepciones

1. En el cambio de contraseña, si las contraseña introducida no es correcta el sistema le mostrará un mensaje de error.

Puntos de extensión

1.

Postcondiciones

1. Se guardan los cambios del usuario.

3.2.9. Planificar Actividades

Caso de Uso: Planificar Actividades

Descripción Este caso de uso permite al gestor de proyectos realizar la planificación de un proyecto en base a las actividades que especifica RUP.

Actor Principal

Gestor de Proyectos

Precondiciones

1. El usuario debe haberse validado en el sistema y ser un Gestor de Proyecto.

Flujo de Eventos

Flujo Básico

1. El sistema le muestra al usuario las fases del proyecto.
2. El usuario elige una fase del proyecto.
3. El sistema le muestra al usuario las iteraciones de dicho fase, y un menú para "Crear Iteración", "Tareas de la Iteración", "Modificar/Eliminar Iteración".

Flujos Alternativos

Crear Iteración

1. El sistema solicita los datos de la nueva iteración.
2. El usuario introduce los datos necesarios.
3. El sistema registra la nueva iteración.

Modificar Iteración

1. El sistema le muestra al usuario la lista de iteraciones.
2. El usuario selecciona una iteración.
3. El sistema muestra la información de dicha iteración.
4. El usuario realiza los cambios pertinentes.

5. El sistema actualiza la información de la iteración en cuestión en el medio de almacenamiento.

Eliminar Iteración

1. El sistema muestra al usuario la lista de iteraciones.
2. El usuario elige una iteración.
3. El sistema muestra información de dicha iteración.
4. El usuario solicita explícitamente borrar dicha iteración.
5. El sistema pide confirmación.
6. El sistema elimina la información de dicha iteración.

Planificar Iteración

1. El sistema solicita que el usuario elija una iteración.
2. El usuario elige una iteración.
3. El sistema presenta la interfaz necesaria para las actividades de gestión.

Excepciones

1.

Puntos de extensión

1.

Postcondiciones

1. La información es actualizada en el medio de almacenamiento.

3.2.10. Caso de Uso Administrar Artefactos

Caso de Uso: Administrar Artefactos

Descripción En este caso de uso se pueden modificar la información relacionada con un artefacto, dar de alta un artefacto o eliminar un artefacto.

Actor Principal

Gestor de Proyecto

Precondiciones

1. Haberse validado en el sistema.

Flujo de Eventos

Flujo Básico

1. El sistema le presenta al usuario la lista de artefactos existentes en el sistema. Para cada artefacto, el usuario tiene la posibilidad de "Ver Detalles", "Modificar" y "Eliminar" y adicionalmente se puede "Agregar Artefacto".
2. El usuario elige la opción "Agregar Artefacto".
3. El sistema le presenta al usuario los campos que debe de llenar para poder agregar un nuevo Artefacto.
4. El usuario introduce la información requerida.
5. Si el usuario decide asociar una plantilla al nuevo artefacto.
   a) El sistema le muestra una pantalla desde donde puede elegir un archivo del sistema de archivo.
   b) El usuario elige el archivo.
   c) El sistema carga el archivo y lo guarda.
6. El sistema graba los cambios y regresa al paso 1.

Flujos Alternativos

Ver Detalles

1. El usuario solicita ver los detalles de un artefacto.

2. El sistema le muestra la información detallada del artefacto especificado.
3. El usuario regresa al paso 1 del Flujo Principal.

Modificar

1. El usuario solicita modificar un artefacto.
2. El sistema le muestra información del artefacto, con la posibilidad de modificarlo.
3. El usuario realiza los cambios pertienentes y solicita al sistema que guarde los cambios realizados.
4. El sistema graba los cambios realizados y regresa al paso 1 del Flujo Principal.

Eliminar

1. El usuario selecciona el artefacto a eliminar.
2. El sistema muestra información detallada de dicho artefacto y espera a que el usuario dé la orden de eliminar.
3. El usuario ordena eliminar el artefacto.
4. El sistema pide confirmación.
5. El usuario confirma.
6. El sistema borra el artefacto.

Excepciones

1. En el subflujo eliminar, si no se puede eliminar un artefacto, el sistema mostrará un mesaje de error al usuario.

Puntos de extensión

1.

Postcondiciones

1. Se guardan los cambios realizados.

3.2.11. Caso de Uso Tareas

Caso de Uso: Tareas

Descripción En este caso de uso el usuario podrá consultar las tareas o actividades que le han sido asignadas. Tendrá la posibilidad de entregar el artefacto generado en su actividad, así como modificar el estado la actividad asignada.

Descripcion Gestor de Proyecto, Usuario

Precondiciones

1. Haberse validado en el sistema.

Flujo de Eventos

Flujo Básico

1. El sistema le muestra al usuario las actividades que le han sido asignadas.
2. El usuario elige una tarea.
3. El sistema le muestra información detallada de la actividad o tarea que tiene asignada, con la posibilidad de modificar información de estatus de la tarea.
4. El usuario realiza los cambios pertienentes.
5. El sistema almacena los cambios y regresa al paso 1.

Flujos Alternativos

Excepciones

1.

Puntos de extensión

1.

Postcondiciones

1. Información de tareas en un estado consistente.

3.3. Desarrollo del Modelo del Dominio

El Modelo del Dominio nos permite identificar los objetos más importantes del sistema. Los objetos, entendiendose como las "cosas" que existen o los eventos que ocurren en el contexto donde el sistema trabaja. Las clases del dominio pueden ser: objetos del negocio que representan cosas que se manipulan en el negocio; objetos del mundo real y conceptos de los que un sistema debe hacer seguimiento y sucesos que ocurren. Despues de realizar el análisis para el proyecto actual, se definió el modelo de la figura 3.3.

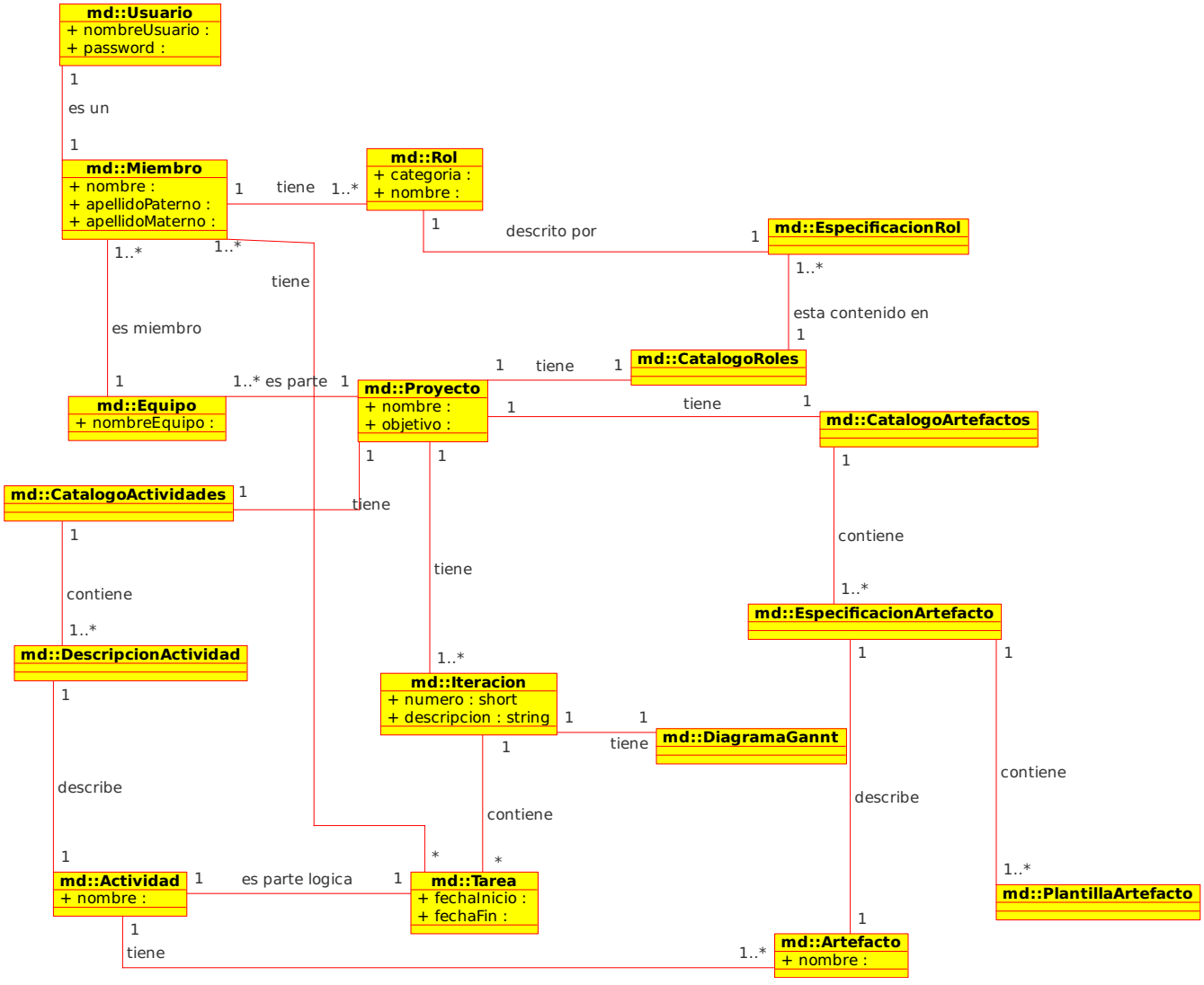

Figura 3.3: Modelo del Dominio

Proyecto: nos referimos al concepto de proyecto como el conjunto de actividades a desarrollar para obtener el software, incluyendo las actividades especificadas por

RUP y los participantes (Analistas, Desarrolladores, Testers, etc.) en el proyecto.

Usuario: Nos referimos a todo miembro del equipo que tendrá acceso al sistema. Visto desde el punto del gestor del proyecto, el usuario es un personal del proyecto.

Rol: Uno de los roles definidos por RUP, el cual es desempeñado por cada personal del proyecto.

Iteración: Una de las iteraciones durante el desarrollo de un proyecto.

Tarea: Una de las actividades a desarrollar durante el transcurso del proyecto.

Catalogo de Actividades: Todas las actividades definidas por RUP, incluyendo su descripción.

Catálogo de Artefactos: Todos los artefactos que indica RUP, con su descripción detallada.

Plantilla: Las plantillas que se definan para cada artefacto en el proyecto.

Diagrama de Gannt: El diagrama de Gannt para cada iteración de acuerdo a las tareas y fechas que el usuario defina.

Capítulo 4

Análisis y Diseño

4.1. Análisis y Diseño

El análisis y diseño es una de las siete disciplinas de RUP y tiene como objetivo:

- Crear un diseño del sistema que cumpla con los requerimientos.
- Desarrollar una arquitectura sólida del sistema.

Los flujos de trabajo que se siguen para lograr lo anterior son:

- Definir la arquitectura del sistema.
- Análisi del comportamiento.
- Diseño de la base de datos.
- Diseño de los componentes del sistema.

Cada uno de estos flujos de trabajo implicar tareas más específicas. A continuación describiremos las tareas realizadas.

4.2. Diseño de Package

Dentro del desarrollo del software orientado a objetos surge la necesidad de colocar las clases creadas en una estructura que permita una distribución apropiada. La manera de lograr lo anterior es agrupando las clases en paquetes de acuerdo a las tareas que realizan. Tomando en cuenta éste criterio, se crearon los siguientes paquetes.

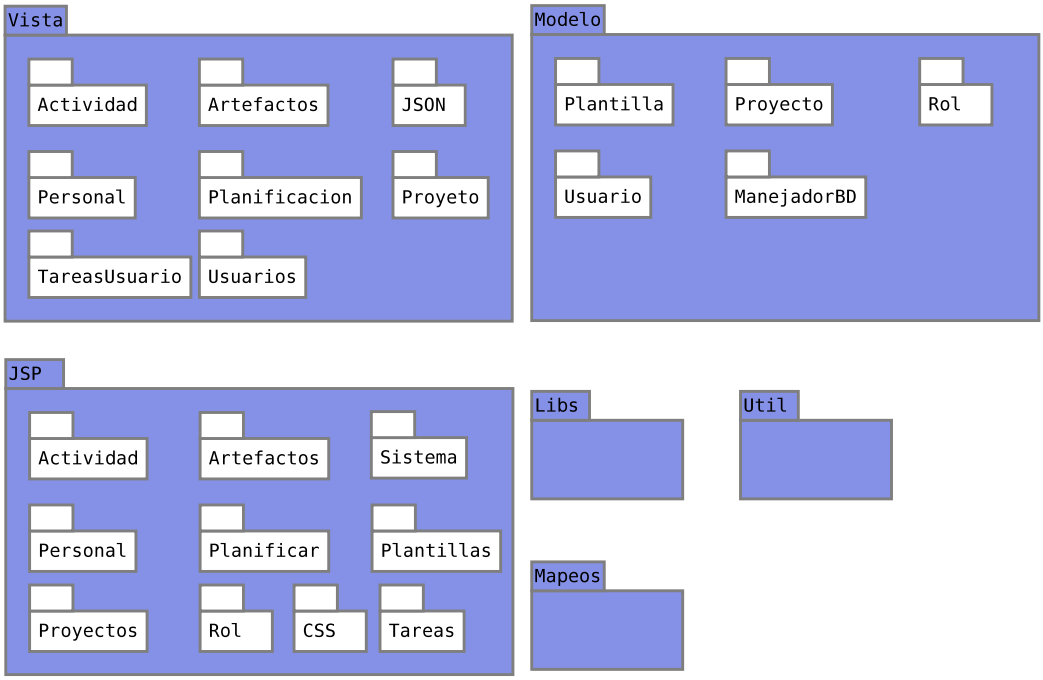

Figura 4.1: Paquetes del sistema

Vista: En este paquete se incluyen todas la clases que manejarán la lógica relacionada con los eventos generados por el usuario. Cada subpaquete se utilizará para una mejor organización de las clases de acuerdo a los eventos que generará el usuario.

JSP: Todas los archivos JSP que serán las interfaces de usuario, organizadas por la clase de actividades realizadas por el usuario.

Modelo: Todas las clases del modelo de negocio, aqui se incluirán las clases persistentes. El subpaquete ManejadorBD contendrá todas las clases que se encargarán de cualquier acceso a la Base de Datos.

Util: Incluiremos todas las clases que se desarrollen y que nos permitan realizar tareas complementarias para la aplicación.

Libs: Todas las clases Java brindadas por terceros utilizadas en la aplicación.

Mapeos: Archivos de mapeos utilizados por Struts.

4.3. Diseño de Subsistemas/Componentes

Para brindar las funcionalidades propuestas para el proyecto, se han definido los siguientes componentes:

Gestión de Datos. Este será el componente por el cual accederemos a los datos persistentes en la BD. Como ya se comentó para su implementación se utilizará Hibernate, implementando todas las clases necesarias para los datos, así como el mapeado a la base de datos usando Annotations.

Gestión de Usuarios: Este será el componente que gestione el acceso de los usuarios al sistema, manejo de sesiones, y todo lo referente a acceso al sistema.

Gestión de Proyectos: Esta es el componente más importante después del de Gestión de Datos. Este componente ofrece todas las funcionalidades propuestas al usuario. Está conformado por un conjunto de Beans que permiten el manejo de la lógica de la aplicación de acuerdo a las solicitudes del usuario.

4.4. Análisis de Clases

Después de haber definido a grandes rasgos los paquetes, los principales componentes del sistema, y definido las funcionalidades que brindará el sistema, así como haber realizado un modelo de objetos del dominio del problema, se está en condiciones de realizar el análisis de clases. Por análisis de clases nos referimos a un conjunto de actividades realizadas para poder definir los principales objetos en el sistema y las responsabilidades que tendrán. Para esto hemos usados el modelo del dominio.

Del modelo de dominio se definieron las sisguientes clases:

Nombre	Proyecto
Descripción	Esta clase tiene como atributos: nombre del proyecto, fecha de inicio y terminación del proyecto, nombre del cliente del proyecto y una descripción del proyecto. Está asociado con una clase Usuario que contiene la información del Gestor de Proyecto, también está relacionado a muchos Usuarios que participan en el proyecto. Además tiene una relación con la clase Fase que representa la diferentes fases del proyecto.

Nombre	Fase
Descripción	Tiene como atributos el nombre de la fase y su descripción. Está relacionado a un Proyecto. Cada fase tiene de cero a n Iteraciones.

Nombre	Iteracion
Descripción	La iteración tiene como atributos un nombre, la fecha de inicio y terminación. Una Iteración tiene de cero a n tareas. La iteración está relacionado a una Fase.

Nombre	Actividad
Descripción	La Actividad contiene la información de una actividad definida por RUP. Tiene como atributos el nombre y la descripción. Una actividad es desempeñada por un usuario que tiene un rol específico. Además cada actividad para ser efectuada necesita de ciertos artefactos, también la actividad al finalizar produce ciertos artefactos.

Nombre	Artefacto
Descripción	Un artefacto es un producto del desarrollo del software. Tiene como atributos un nombre, una descripción, un objetivo y una representación UML, cuando aplique. El artefacto está asociado a un Rol.

Nombre	Disciplina
Descripción	Una de las 7 disciplinas que define RUP. Tiene como atributos el nombre y la descripción.

Nombre	Plantilla
Descripción	La plantilla es un formato de documento para realizar la documentación de cierto artefacto. Tiene como atributos el nombre y la fecha de creación. Está relacionada a un Proyecto y a un Artefacto.

Nombre	Rol
Descripción	Uno de los roles definidos por RUP. Sus atributos son el nombre, la descripción y la categoría a la cual pertenece.

Nombre	Usuario
Descripción	El usuario es un participante del proyecto. Tiene como atributos el nombre, apellido paterno, apellido materno, correo, telefono, login y password. El Usuario está relacionado a un Proyecto, desempeña uno o más Roles y tiene de cero a n Tareas.

Nombre	EjemplarArtefacto
Descripción	Es un Artefacto realizado para un proyecto específico. Sus atributos son el nombre, fecha de creación y contenido. Está asociado a un Artefacto y a un Proyecto.

Nombre	Tarea
Descripción	Es un trabajo que se ha asignado a un participante del proyecto. Tiene como atributos el nombre, la fecha de inicio y terminación, el porcentaje realizado. Está asociado a uno o muchos Usuarios, está relacionada con una Actividad. Además la Tarea puede realizarse después de que finalizen otras Tareas. La Tarea puede tener subtareas.

Estás son clases que nos van a permitir manejar información con respecto a los principales objetos del problema. Sin embargo, se van a utilizar otras clases que nos permitan el acceso a la base de datos, ya sea para recuperar registros de la base de datos o realizar consultas muy específicas. A continuacion listamos la clases definidas:

Nombre	ManejadorProyecto
Descripción	Esta clase nos permitirá realizar las operaciones de inserción, actualización y borrado de un proyecto. También podremos realizar consultas de las fases e iteraciones del proyecto así como los usuarios que participan en el proyecto,

Nombre	ManejadorFase
Descripción	Esta clase tendrá las operaciones de inserción, actualización y borrado de una fase.

Nombre	ManejadorIteracion
Descripción	Esta clase nos permitirá hacer inserciones de registros de iteraciones, recuperar iteraciones de cierto proyecto, una iteración específica o las tareas de una iteración, además de la eliminación o modificación de iteraciones en la base de datos.

Nombre	ManejadorActividad
Descripción	Esta clase debe permitirnos realizar operaciones de inserción, actualización y borrado de Actividades, nos permitirá realizar búsquedas tales como las actividades que pertenecen a una disciplina, los artefactos requeridos o producidos por una actividad, buscar una actividad en específico, y obtener los artefactos de una actividad.

Nombre	ManejadorArtefacto
Descripción	Nos permitirá realizar búsqueda de artefactos basados en su nombre, actualización, inserción y borrado de un Artefacto.

Nombre	ManejadorDisciplina
Descripción	Nos permitirá realizar búsqueda de una disciplina basado en su identificador, además de la inserción, actualización y eliminación.

Nombre	ManejadorRol
Descripción	Búsqueda del rol basado en su nombre e inserción, actualización y eliminación de un Rol.

Nombre	ManejadorUsuario
Descripción	Esta clase nos permitirá realizar las inserciones, actualización y eliminación de usuarios en la Base de Datos. También se podrá buscar los usuarios que pertenecen a un proyecto en específico, los usuarios que no participan en un proyecto, recuperar todos los usuarios que participan en un proyecto específico y que tienen cierto rol, hacer busqueda de usuarios y recuperar los proyectos en los que participa un usuario.

Nombre	ManejadorTarea
Descripción	La clase nos permitirá consultar las tareas de un proyecto en específico, recuperar las tareas que preceden o son subtareas de una tarea, buscar una tarea en específico y recuperar los usuarios que estan asignados a una tarea.

La idea de lo anterior es que cada una de las clases nos permita manejar la informacion relacionada a una clase persistente. De ahí el nombre Manejador<NombreClase>. Todas estas clases en su conjunto conformarán el componente “Gestión de Datos”.

Por otra parte de acuerdo al MVC explicado anteriormente, la vista(ejem. interfaz de usuario) está asociada a un Bean. Estos Beans en su conjunto conforman lo que es el controlador, responsable de gestionar los eventos de entrada desde la vista. Lo

cual comentaremos con más detalle en el capítulo 8.

4.5. Diseño de Clases

Una clase en Programación Orientado a Objetos es una abstracción que representa un conjunto de objetos con un comportamiento e interfaz común. Es decir, se refiere a un conjunto de cosas (fisicas o abstractas) que tienen comportamiento y característica común. Una clase tiene atributos que describen el estado del objeto, mientras que los métodos es la forma de representar el comportamiento de dicho objeto.

Después de haber identificado las principales clases que tendrá el sistema, nos ocuparemos de describirlas más detalladamente, incluyendo los atributos y métodos que tendrá. Después de esta actividad obtuvimos el diagrama mostrado en la figura 4.2.

Como se puede apreciar, todas las clases tienen una propiedad identificadora. En algunos casos es un identificador simple y en otros es compuesto. Este requerimiento no es indispensable, pero lo incluimos porque deseamos aprovechar las funcionalidades que nos provee Hibernate. Todas estas clases serán persistentes, es decir, se almanecarán en su momento en la base de datos.

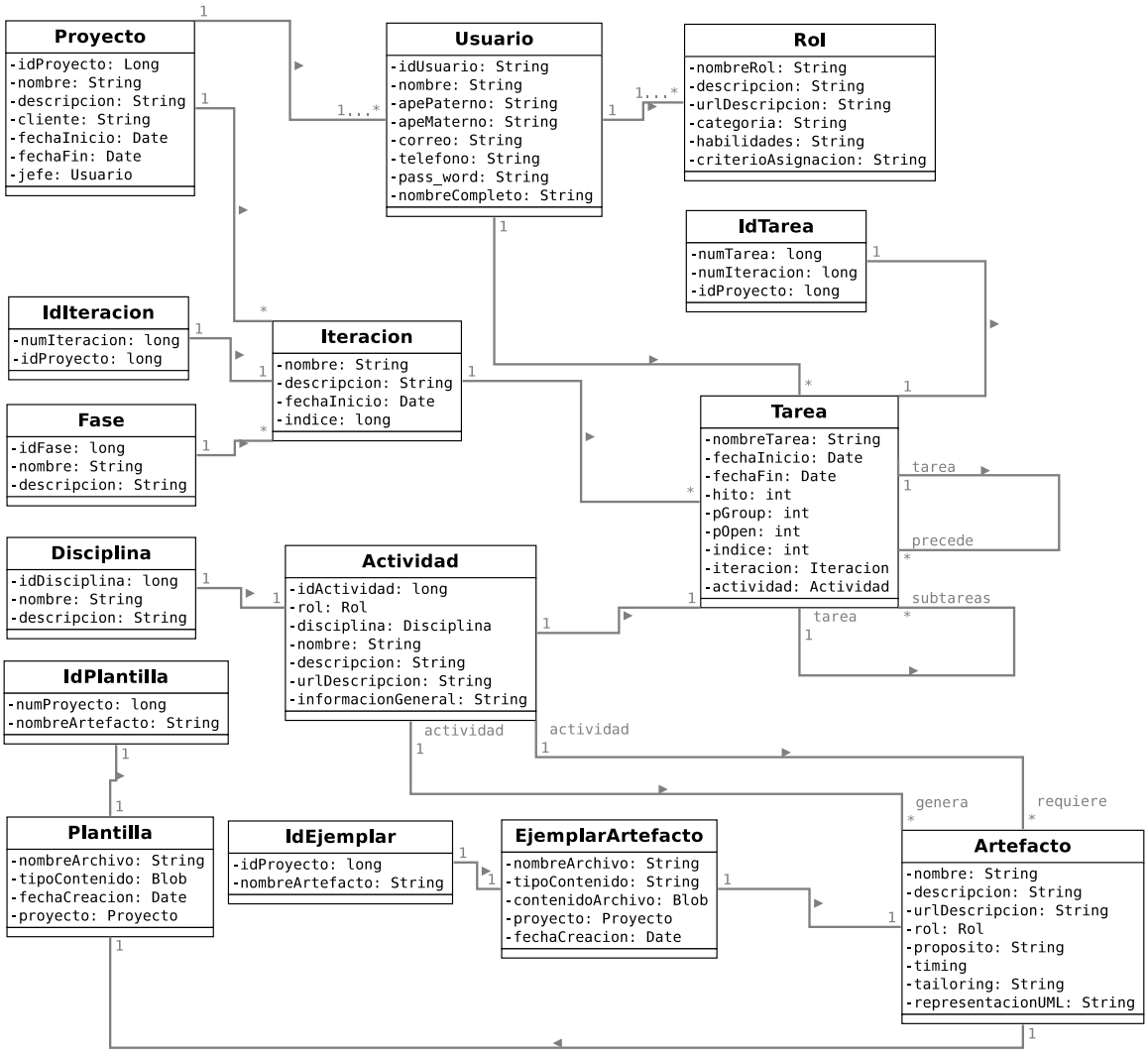

Figura 4.2: Diagrama de clases persistentes.

.

Posterior a la identificación de clases persistentes, ahora identificamos las clases necesarias para tener acceso a los datos. Éstas clases se aprecian en la figura 4.3. Todas tienen los método necesarios para realizar las operaciones de borrado, actualización e inserción de lass clases persistentes mapeados a la Base de Datos, pero nos los mostramos por claridad.

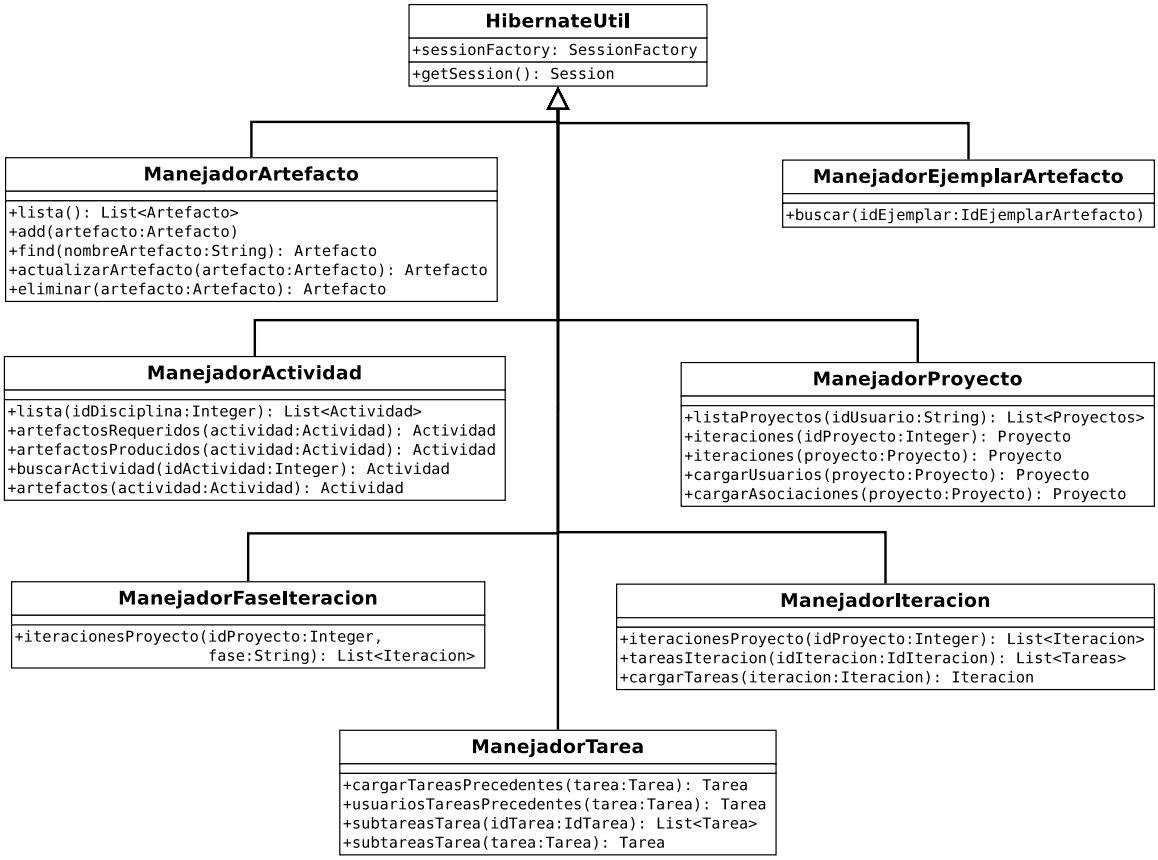

Figura 4.3: Diagrama de clases DAO.

Por otra parte, tenemos que crear una serie de clases que se encargarán de la lógica de la aplicación. Dichas clases conforman parte de lo que es el modelo de acuerdo al MVC. Generalmente se crea una clase por cada pantalla, un clase que manejará toda la lógica relacionada con los eventos generados desde esa pantalla por el usuario. El criterio que seguiremos para nombrar dichas clases será el siguiente:

- Cada pantalla tendrá un nombre *nombre_pantalla* el cual se utilizará para nombrar a la clase, agregando la terminación *Action*, por lo que la clase tendrá el nombre: *nombre_pantallaAction.java*.

Se crearán tantas clases como pantallas existan, a menos que se pueda reutilizar una existente. Todo lo anterior se explica más detalladamente en el capítulo 7.

4.6. Diseño del Modelo de Datos

El presente sistema es un prototipo para gestionar proyectos de software que se desarrollen bajo el modelo de desarrollo RUP (Rational Unified Process), por lo que debe cumplir con los siguientes requisitos.

Cada proyecto se identifica por un código identificador, un nombre, una descripción breve, fecha de inicio del proyecto y fecha de finalización, además del nombre del cliente que solicita el proyecto.

Cada proyecto tiene un único Jefe de Proyecto y puede haber uno o más participantes en el proyecto. Cada participante puede desempeñar más de un rol.

El participante del proyecto tiene un identificador único, un nombre, apellido paterno, apellido materno, correo, teléfono, nombre de usuario y contraseña.

Cada rol tiene un nombre que lo identifica de forma única de los demás roles, una descripción breve, una url de un archivo html que contiene la descripción detallada del mismo.

Cada proyecto tiene cuatro fases: Inicio, Elaboración, Construcción y Transición. Cada fase cuenta con una descripción breve, fecha de inicio, fecha de finalización.

Cada fase contiene una o más iteraciones. Cada iteración tiene un número de secuencia, una descripción del objetivo de dicha iteración, la fecha de inicio y fecha de finalización.

Cada iteración cuenta con un conjunto de tareas a realizar. La tarea tiene un identificador, un nombre, una fecha de inicio, fecha de finalización, los participantes que realizarán la tarea, y la actividad específica del modelo de desarrollo que se realizará en la tarea. Todas las actividades las define el modelo de desarrollo RUP. La tarea tiene un estado que puede ser en curso o finalizado.

Las actividades tienen un identificador, un nombre, una URL que tiene la ubicación del HTML que describe de forma detallada dicha actividad, una descripción breve, los artefactos de entrada y de salida, así como la disciplina a la que pertenece la actividad. Las disciplinas las define el modelo de desarrollo RUP y son: Modelado del Negocio, Requerimientos, Análisis y Diseño, Implementación, Test, Despliegue, Administración de Cambios y Configuración, Administración del Proyecto, Ambiente de Desarollo.

Los artefactos también los define RUP, los define para cada disciplina. Son alrededor de 60 artefactos. Cada artefacto tiene un identificador, una descripción breve, una url del archivo html que lo describe completamente, un conjunto de actividades donde el artefacto sirve de entrada y un conjunto de actividades donde el artefacto es una salida.

Cada disciplina tiene un identificador, un nombre, una descripción, un conjunto de actividades que pertenecen a la disciplina, el conjunto de artefactos que pertenecen a la disciplina.

Cada artefacto puede tener una o más plantillas, donde la plantilla es el formato para entregar el artefacto.

La plantilla tiene un identificador, un nombre y una ubicacion url.

4.7. Modelo Entidad Relación

Después de hacer una descripción de las diversas entidades que serán necesarias, ahora pasamos a realizar el Modelo Entidad Relación, por lo que a continuación iremos mostrando las entidades mencionadas, mostrando la relación entre las mismas.

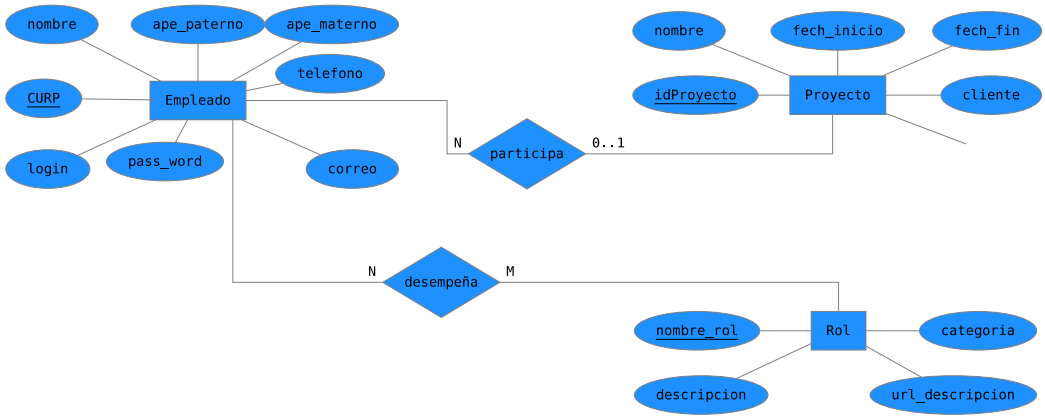

Figura 4.4: Empleado, Proyecto y Rol

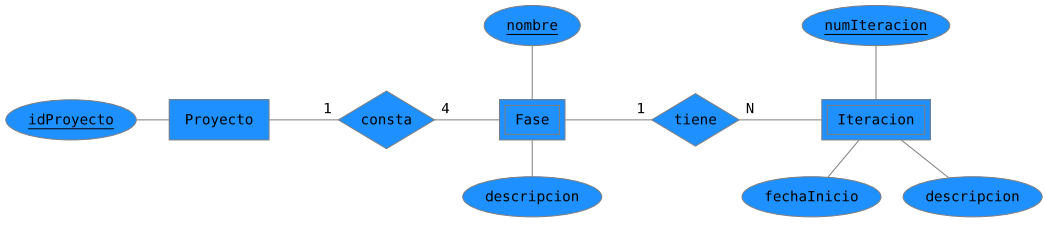

Figura 4.5: Artefacto, Plantilla, Actividad, Disciplina

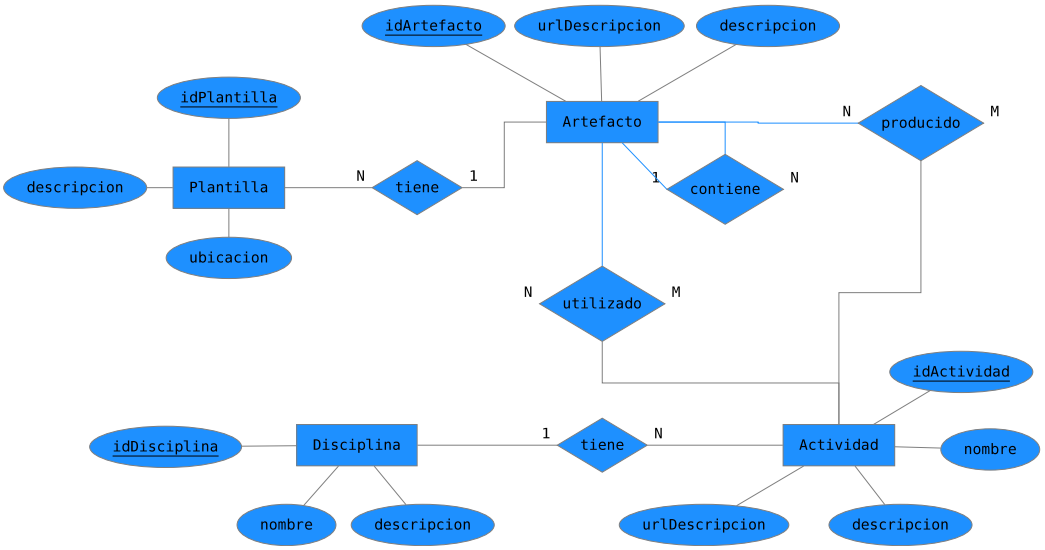

Figura 4.6: Artefacto, Plantilla, Actividad, Disciplina

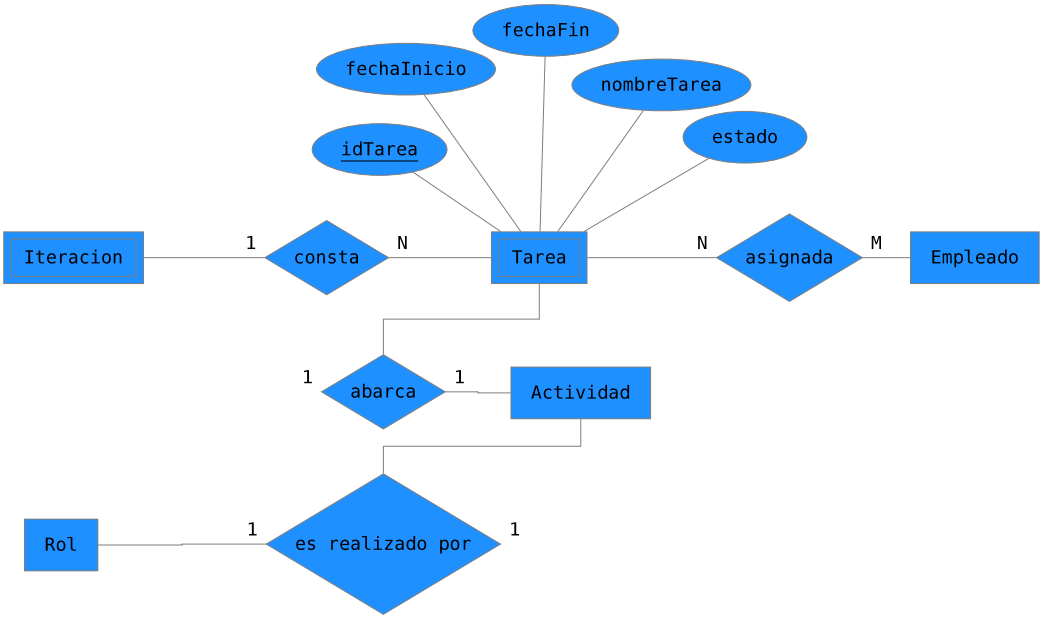

Figura 4.7: Tarea, Iteracion

4.8. Normalización

Toda Base de Datos debe de pasar por un proceso llamado Normalización para que esté bien construida. La normalización son una serie de pasos que se siguen a fin de tranformar datos complejos a un conjunto de estructuras de datos más pequeñas, que además de ser más simples y estables, son más fáciles de mantener. La normalización ayuda reducir los problemas de lógica en la BD, elimina redundancia de datos que pudiera haber, permite llegar a una estructura de datos más eficiente y es más fácil manejar inserciones, actualizaciones y borrados en la BD.

Existen básicamente tres niveles de normalización: Primera Forma Normal (1FN), Segunda Forma Normal (2NF) y Tercera Forma Normal(3NF). Cada uno de los grados de normalización tiene sus reglas.

Primera Forma Normal (1FN): Una relación está en 1FN si y sólo si cada atributo tiene un valor sencillo para cada tupla. Esto significa que el domino de los atributos debe ser atómico, es decir, los elementos del dominio son unidades indivisibles.

Segunda Forma Normal (2FN): Una relación está en segunda forma normal si y sólo si está en Primera Forma Normal y todos los atributos no clave son completamente dependientes funcionales sobre la clave.

Dependencia Completa Funcional: En una relación R, el atributo A de R es completamente dependientes funcional sobre un atributo o conjunto de atributos X de R si A es funcionalmente dependiente sobre X pero no es funcionalmente dependiente sobre cualquier subconjunto propio de X.

Tercera Forma Normal (3FN): Una relación está en tercera forma normal si, siempre que exista una dependencia funcional no trivial $X \rightarrow A$, entonces o X es una superclave o A es un miembro de alguna clave candidata. En otras palabras, cada atributo no clave debe depender de la clave, toda la clave y nada más de la clave.

Tercera Forma Normal Boyce-Codd (FNBC): Una relación está en forma FNBC si, siempre que exista una dependencia funcional no trivial $X \rightarrow A$, entonces X es una superclave. Es decir, todos los determinantes son superclaves.

El grado de normalización al cual llegar en una BD queda a criterio del diseñador, dependiendo de las necesidades del negocio. Por ejemplo, en ocasiones quizá se desee dejar cierta rendundancia de información en relaciones sobre las cuales se sabe que se harán muchas consultas.

Para nuestra estructura de BD llegaremos a lo mucho a FNBC en el caso de algunas relaciones.

4.8.1. Usuario

La tabla se puede ver en la figura 4.8.

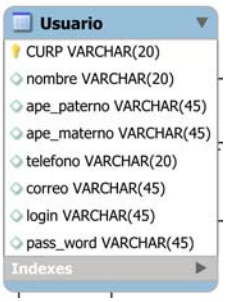

Figura 4.8: Tabla Usuario

Identificar Dependencias Funcionales

- CURP → nombre, ape_paterno, ape_materno, telefono, correo, login, pass_word

1FN

Todos los atributos tienen valores simples por lo que está en 1FN.

2FN

La dependencia identificada:

- CURP → nombre, ape_paterno, ape_materno, telefono, correo, login, pass_word

Por lo tanto, todos los atributos no clave dependen completamente de la clave.

FNBC

El determinante es clave, por lo que cumple con FNBC.

4.8.2. Rol

La tabla se muestra en la figura 4.9.

Figura 4.9: Tabla Rol

Dependencias Funcionales

La dependencia funcional es:

- nombreRol → descripcion, urlDescripcion, categoria

1FN

Todos los atributos tienen valores simples por lo que está en 1FN.

2FN

Todos los atributos no clave dependen completamente de la clave.

FNBC

El determinante es clave, por lo que cumple con FNBC.

4.8.3. Proyecto

La tabla se observa en la figura 4.10.

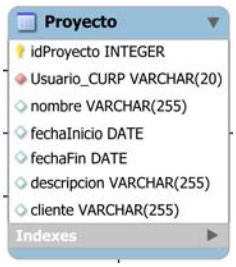

Figura 4.10: Tabla Proyecto

Dependencias Funcionales

La dependencia funcional es:

- idProyecto → Usuario_CURP, nombre, fechaInicio, fechaFin, descripcion, cliente

1FN

Todos los atributos tienen valores simples por lo que está en 1FN.

2FN

Todos los atributos no clave dependen completamente de la clave.

FNBC

El determinante es clave, por lo que cumple con FNBC.

4.8.4. Fase

La tabla se muestra en 4.11.

Figura 4.11: Tabla Fase

Dependencias Funcionales

La dependencia funcional es:

- idFase → nombre, descripcion

1FN

Todos los atributos tienen valores simples por lo que está en 1FN.

2FN

Todos los atributos no clave dependen completamente de la clave.

FNBC

El determinante es clave, por lo que cumple con FNBC.

4.8.5. Iteración

En 4.12 puede observar la tabla.

Figura 4.12: Tabla Iteracion

Dependencias Funcionales

La dependencia funcional es:

- numIteracion, Proyecto_idProyecto → fechaInicio, descripcion

1FN

Todos los atributos tienen valores simples por lo que está en 1FN.

2FN

Todos los atributos no clave dependen completamente de la clave.

FNBC

El determinante es clave, por lo que cumple con FNBC.

4.8.6. Tarea

Ver tabla en figura 4.13.

Figura 4.13: Tabla Tarea

Dependencias Funcionales

La dependencia funcional es:

- numTarea,Iteracion_numIteracion,Iteracion_Proyecto_idProyecto → Actividad_idActividad, nombreTarea, fechaInicio, fechaFin, estado

1FN

Todos los atributos tienen valores simples por lo que está en 1FN.

2FN

Todos los atributos no clave dependen completamente de la clave.

FNBC

El determinante es clave, por lo que cumple con FNBC.

4.8.7. Actividad

Ver tabla en figura 4.14.

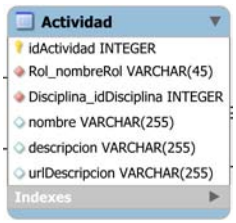

Figura 4.14: Tabla Actividad

Dependencias Funcionales

La dependencia funcional es:

- idActividad → Rol_nombreRol, Disciplina_idDisciplina, nombre, descripcion, urlDescripcion

1FN

Todos los atributos tienen valores simples por lo que está en 1FN.

2FN

Todos los atributos no clave dependen completamente de la clave.

FNBC

El determinante es clave, por lo que cumple con FNBC.

4.8.8. Disciplina

Ver tabla en figura 4.15.

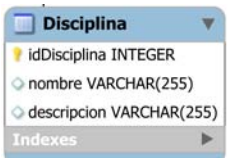

Figura 4.15: Tabla Disciplina

Dependencias Funcionales

La dependencia funcional es:

- idDisciplina → nombre, descripcion

1FN

Todos los atributos tienen valores simples por lo que está en 1FN.

2FN

Todos los atributos no clave dependen completamente de la clave.

FNBC

El determinante es clave, por lo que cumple con FNBC.

4.8.9. Artefacto

Ver tabla en figura 4.16.

Dependencias Funcionales

La dependencia funcional es:

Figura 4.16: Tabla Artefacto

- nombre → descripcion, urlDescripcion, content, fechaCreado, contentType, nombreArchivo, tamanoArchivo

1FN

Todos los atributos tienen valores simples por lo que está en 1FN.

2FN

Todos los atributos no clave dependen completamente de la clave.

FNBC

El determinante es clave, por lo que cumple con FNBC.

4.8.10. Plantilla

Ver tabla en figura 4.17.

Figura 4.17: Tabla Plantilla

Dependencias Funcionales

La dependencia funcional es:

- Proyecto_idProyecto, Artefacto_nombre → nombreArchivo, tipoContenido, contenido, fechaCreado

1FN

Todos los atributos tienen valores simples por lo que está en 1FN.

2FN

Todos los atributos no clave dependen completamente de la clave.

FNBC

El determinante es clave, por lo que cumple con FNBC.

4.8.11. Ejemplar

Ver tabla en figura 4.18.

Figura 4.18: Tabla Ejemplar

Dependencias Funcionales

La dependencia funcional es:

- Proyecto_idProyecto, Artefacto_nombre → nombreArchiv, tipoContenido, contenidoArchivo

1FN

Todos los atributos tienen valores simples por lo que está en 1FN.

2FN

Todos los atributos no clave dependen completamente de la clave.

FNBC

El determinante es clave, por lo que cumple con FNBC.

Como se pudo apreciar, la BD, está normalizada a un grando razonable para el uso que se le dará. Lo anterior se debe a que se siguió un método para el diseño de la BD y a que su tamaño no es tan grande y complejo.

4.9. Paso del Modelo ER al diseño de la Base de Datos

4.9.1. Base de Datos

Por el momento sólo mostraremos el diagrama de la base de datos completa, posteriormente en el capitulo 5, explicaremos los detalles de la creación de la base de datos. Las tablas de la base de datos se pueden observar en la figura 4.19.

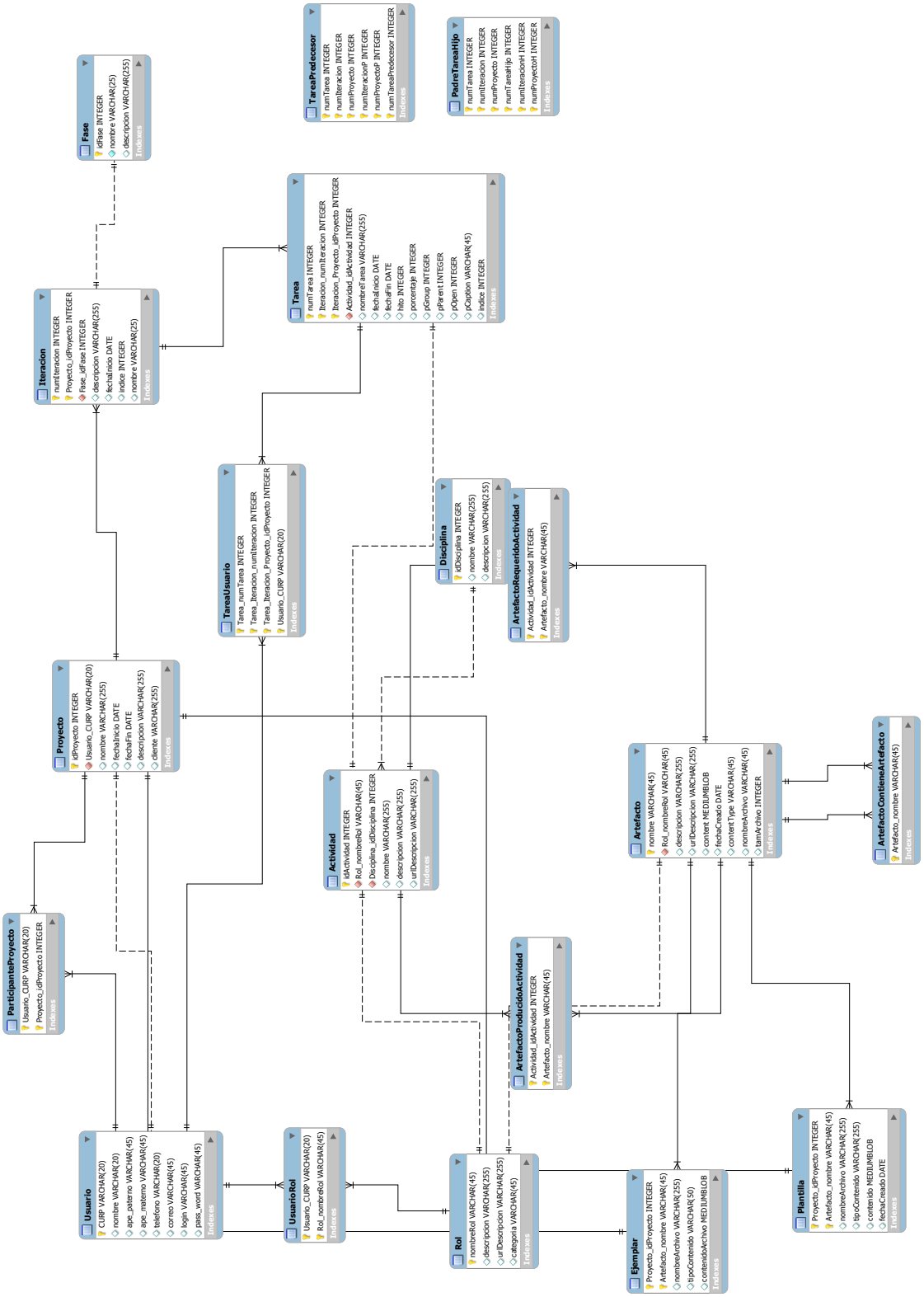

Figura 4.19: Base de Datos

4.10. Mapa de Navegación

El mapa de navegación es la representación de cómo estará estructurada nuestra aplicación web. Nos permite tener un panorama general de las pantallas del sistema. Hemos realizado el mapa de navegación agrupando en secciones tareas similares. De acuerdo a lo anterior hemos establecido las siguientes secciones: Inicio, Proyectos, Personal, Planificar, Actividades, Rol y Artefactos. Estos en la aplicación serán las opciones del menú principal. Cada opción del menú estará asociada a una pantalla.

En la figura 4.20 se pueden ver las secciones u opciones del menú principal.

Figura 4.20: Menú Principal

Ahora desglosaremos cada uno de las opciones del menú.

4.10.1. Inicio

En esta opción el usuario podrá tener los campos necesarios para poder validarse en el sistema, así como una descripción breve del sistema. Al ser validado correctamente el sistema lo situará automáticamente en la pantalla asociada a la opción Proyectos del menú principal.

4.10.2. Proyectos

En la pantalla asociada a la opción Proyectos el usuario podrá realizar lo siguiente: crear, modificar o eliminar un proyecto, visualizar los proyectos en los que participa, elegir un proyecto (sobre el cual se realizarán todas las acciones que el usuario haga posteriormente), visualizar detalles de un proyecto. En la figura 4.21 se aprecia la navegación.

4.10.3. Personal

En la pantalla asociada a la opción Personal, el usuario dará de alta nuevos usuarios en el sistema, agregar personal a un proyecto, visualizar información detallada

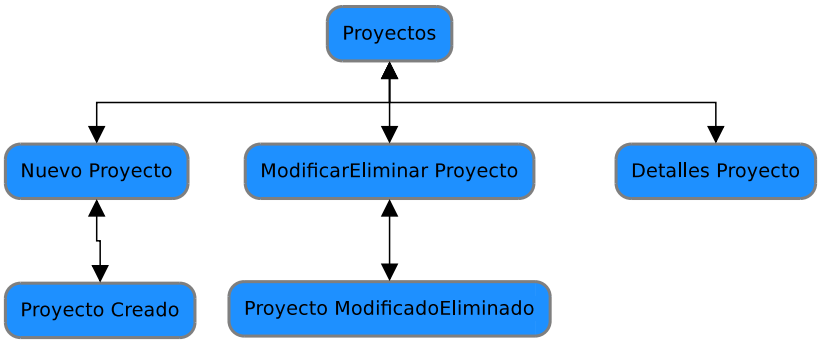

Figura 4.21: Navegación en Proyectos

de un usuario, modificar o eliminar un usuario. En la figura 4.22 se observa el mapa de navegación para esta opción.

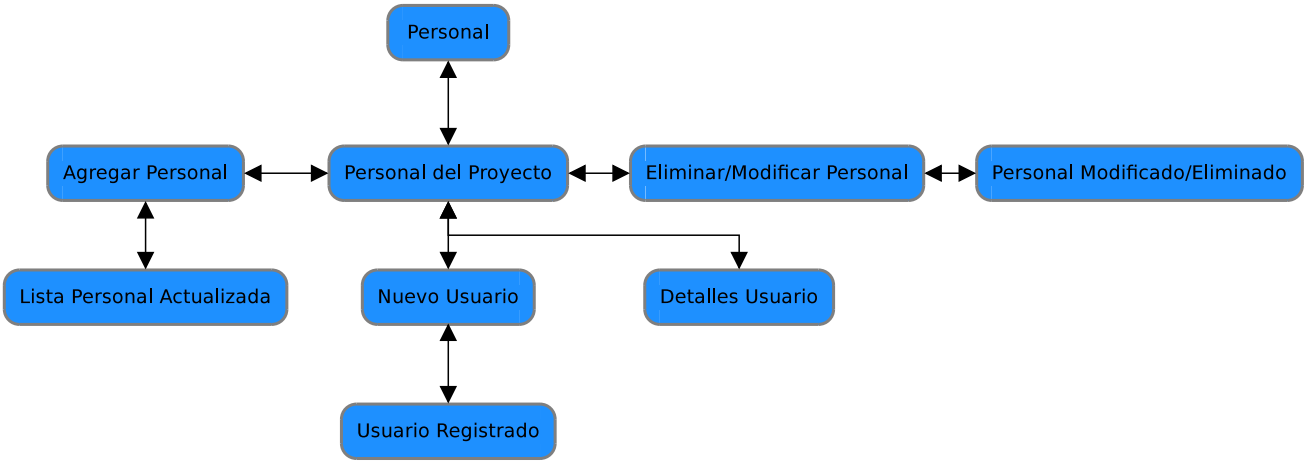

Figura 4.22: Navegación en Personal

4.10.4. Planificar

En la opción Planificar, el usuario podrá crear iteraciones según RUP, dar de alta tareas, modificarlas o eliminarlas, así como asignarlas. Tendrá la opción de elegir la actividad a la que será asociada cada tarea. En la figura 4.23 se muestra el mapa de navegación.

4.10.5. Plantillas

En Plantillas se permitirá al usuario agregar un formato de documento maestro para cada artefacto establecido por RUP, el cual servirá como guía para el personal

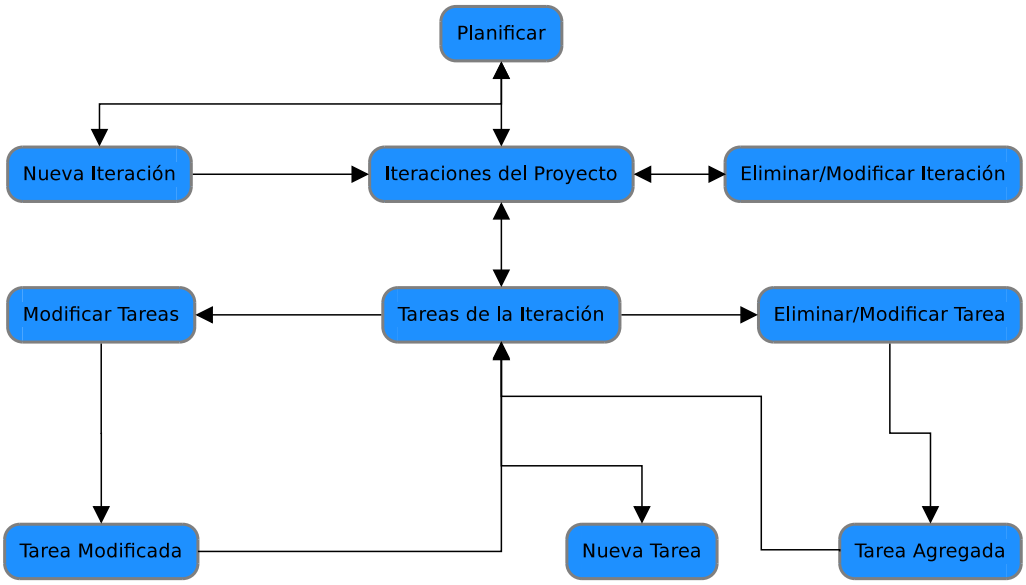

Figura 4.23: Navegación en Planificar

que le corresponda realizar un entregable de dicho artefacto. La navegación propuesta se ve en la figura 4.24.

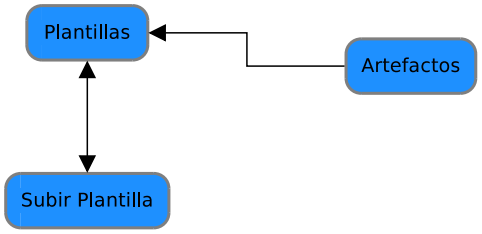

Figura 4.24: Navegación en Plantillas

4.10.6. Actividades

Las actividades son parte fundamental de RUP, por ello, en la opción Plantillas se darán alternativas al usuario para gestionarlas de acuerdo a las necesidades de un proyecto en particular. Podrá asociar las actividades que desee a su proyecto. Para ello se deberá contar con un catálogo de actividades definidas por RUP. En esta opción tiene la posibilidad de visualizar información detallada de cada actividad, tales como, artefactos asociados, rol de quién se le pueden asignar, y otra información relacionada. En la figura 4.25 presentamos el mapa de navegación.

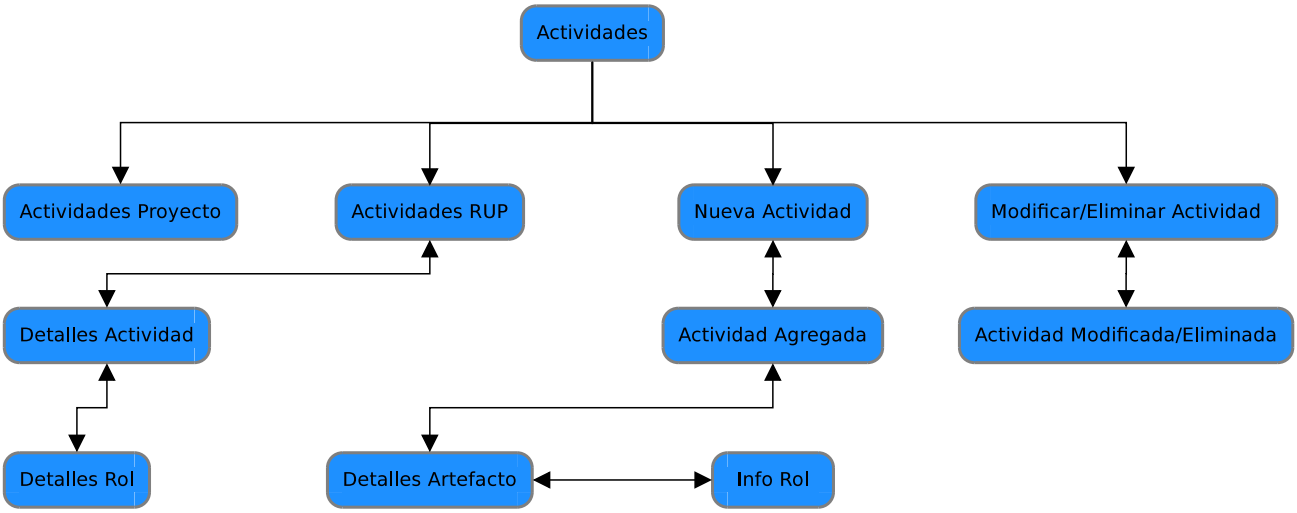

Figura 4.25: Navegación en Actividades

4.10.7. Tareas

Una de las partes más importantes de un proyecto es la asignación de actividades a cada miembro del equipo del proyecto. En esta opción, el usuario podrá gestionar las tareas asignadas a cada miembro del equipo. Para cada tarea se puede asociar una actividad definida por RUP, seleccionar los entregables (Artefactos), fechas para las entregas, entre otros. En la figura 4.26 mostramos la forma de navegación.

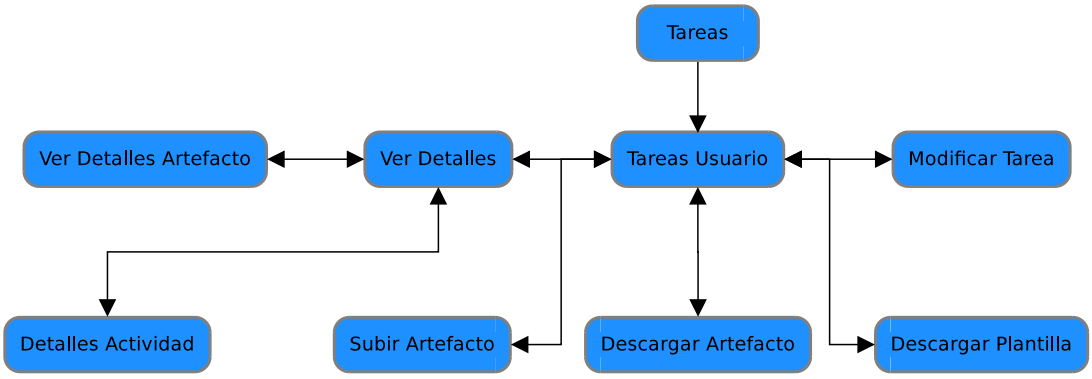

Figura 4.26: Navegación en Tareas

Esta es de forma general la estructura que tendrá el sistema, posteriormente comentaremos sobre la implementación de las pantallas.

Capítulo 5

Base de Datos

5.0.8. Creación de la Base de Datos

Aunque ya mostramos una imagen de las tablas creadas en la base de datos, en este capítulo mencionaremos aspectos importantes que se tomaron en cuenta al momento de implementar la base de datos con MySQL. Cabe mencionar que no se mostrará todo el código SQL, sino sólo las porciones que consideramos pertinentes.

Comenzaremos mencionando que gran parte de la creación de la base de datos completa implica crear las tablas que hemos definido. Sin embargo, antes se tiene que crear la base de datos. En el lenguaje SQL para crear una base de datos se utiliza la sentencia: *CREATE DATABASE nombre_base_datos*. En nuestro caso le hemos puesto el siguiente nombre: *PROTOTIPO_WEB*, para ello hemos ejecutado la sentencia:

CREATE DATABASE PROTOTIPO_WEB

Posteriormente se tiene que crear al menos un usuario, el cual se hace con la sentencia:

CREATE USER user [IDENTIFIED BY [PASSWORD] 'password'] [, user [IDENTIFIED BY [PASSWORD] 'password']] ...

Hemos creado el siguiente usuario:
CREATE USER prototipo IDENTIFIED BY 'aplicacion_01';

Ahora le daremos permisos sobre la base de datos *PROTOTIPO_WEB*, con la sentencia:

GRANT ALL ON prototipov6. TO prototipo;*

Con lo anterior ya hemos creado la base de datos y otorgados los permisos al usuario *prototipo*. Ahora bien, al momento de crear las tablas surge la necesidad de manejar lo que se conoce como integridad referecial, el cual comentaremos en la siguiete sección.

5.0.9. Integridad Referencial

La integridad referencial implica que una clave foránea de una tabla siempre debe aludir a un registro válido en la tabla a la que haga referencia. Esto garantiza que la relación entre las dos tablas siempre sea consistente. Una inconsistencia se presenta cuando una clave foránea hace referencia a una clave primaria que no existen en la tabla referenciada. Por ello, el uso de restricciones de integridad referencial permite que se efectuén o no las operaciones de actualización y eliminación.

La implementación de estas restricciones de integridad referencial las podemos hacer desde la lógica de programación. Sin embargo, es mucho mejor que el propio motor de la base de datos las maneje. En MySQL 3.23.44 y posteriores, el motor InnoBD(un motor de almacenamiento transaccional) soporta chequeo de restricciones de claves foráneas, incluyendo *CASCADE, ON DELETE y ON UPDATE*. Algunos de los beneficios de estas restricciones son[1]:

- Las restricciones de clave foránea hacen más dificil que un programador introduzca inconsistencias en la base de datos.
- Usando actualizaciones y borrados en cascada puede simplificarse el código de aplicación.
- Reglas diseñadas correctamente para claves foráneas pueden ayudar a la documentación de las relaciones de las tablas.

Por lo anterior hemos decidido manejar la integridad referencial desde la base de datos y no desde la lógica de la aplicacón. Si bien es cierto que esto agrega procesamiento extra a la base de datos, debido a todos los chequeos que tiene que realizar, para nuestra aplicación eso no es muy importante por el momento.

Para agregar una restricción usando MySQL se usa la sentencia *CONSTRAINT* que tiene la sintaxis siguiente:

[1]http://dev.mysql.com/doc/refman/5.0/es/ansi-diff-foreign-keys.html

```
[CONSTRAINT [symbol]] FOREIGN KEY
    [index_name] (index_col_name, ...)
    REFERENCES tbl_name (index_col_name,...)
    [ON DELETE reference_option]
    [ON UPDATE reference_option]

reference_option:
    RESTRICT | CASCADE | SET NULL | NO ACTION
```

A continuación describimos los posibles valores:

- *symbol* Nombre de la restricción.
- *index_name*
- *index_col_name* Nombre(s) de las columnas que conforman la llave primaria o llave foránea.
- *tbl_name* Nombre de la tabla a la que referencia la llave foránea.
- *reference_option* Una de las opciones:
  - RESTRICT No permite operaciones de actualización y borrado en la tabla maestra.
  - CASCADE Actualiza o elimina todos los registros de la tabla referencia cuya llave primaria sea la llave foránea de la tabla maestra.
  - SET NULL Al actualizar o elimina el registro de la tabla maestra, establece a null la llave foránea.
  - NO ACTION La llave primaria de la tabla referenciada no se puede actualizar o eliminar si es llave foránea en una tabla maestra.

5.0.10. Tabla Rol

Esta tabla contiene registros de un rol de acuerdo a la especificación de RUP. Para esta tabla definimos las siguientes restricciones:

- La llave primaria es *nombreRol* que es de tipo *VARCHAR*

Si esta tabla fuese utilizada con mucha frecuencia, los más recomendable sería utilizar una clave primaria de tipo *Integer*, ya que eso agilizaría las consultas. Sin embargo, debido a que no es con frecuencia utilizada y no contendrá muchos registros (no más de 50) hemos decidido utilizar el nombre del rol como llave primaria.

5.0.11. Tabla Disciplina

La Disciplina es un concepto que meneja RUP, para el cual almacenaremos la información asociada en esta tabla. Nuestras restricciones son:

- La llave primaria es la columna *idDisciplina* de tipo *Integer*.

5.0.12. Tabla Fase

RUP define cuatro fases, que son las siguientes:

- Inicio
- Elaboración
- Construcción
- Transición

Para esta tabla se tienen las siguientes restricciones:

- La llave primaria será la columna *idFase*, de tipo *Integer*.
- El nombre de la fase no se puede repetir.

```
ALTER TABLE Fase ADD CONSTRAINT NOMBREFASE UNIQUE(nombre)
```

5.0.13. Tabla Usuario

Esta tabla almacenará la información de un usuario del sistema y en consecuencia participante de algún proyecto. Las restricciones que hemos establecido para esta tabla son:

- La llave primaria es la columna *CURP*.
- La columna correo no puede repetirse.

```
ALTER TABLE Usuario ADD CONSTRAINT UNIQUE(correo);
```

- El passwor del usuario estará encriptada con el algoritmo MD5.

Tabla UsuarioRol

Un usuario puede tener muchos roles, para ello se ha creado la tabla *UsuarioRol*, que tiene los siguientes campos con las restricciones que se mencionan:

- *Usuario_CURP, Rol_nombreRol*. Conforman la llave primaria, de tipo String.

```
PRIMARY KEY (‘Usuario_CURP’,‘Rol_nombreRol’)
```

- *Usuario_CURP*. Llave foránea que referencia a la columna *CURP* en la tabla *Usuario*. Al eliminarse o actualizarse deben de realizarse la mismas operaciones en la tabla.

```
CONSTRAINT ‘UsuarioRol_fk_curp’
FOREIGN KEY (‘Usuario_CURP’)
REFERENCES ‘Usuario’ (‘CURP’)
ON DELETE CASCADE ON UPDATE CASCADE
```

- *Rol_nombreRol*. Llave foránea que referencia a la columna *nombreRol* de la tabla *Rol*. Al actualizarse en la tabla *Rol* se debe de actualizar también en *UsuarioRol*.

```
CONSTRAINT ‘UsuarioRol_fk_nombreRol’ FOREIGN KEY (‘Rol_nombreRol’)
REFERENCES ‘Rol’ (‘nombreRol’) ON UPDATE CASCADE
```

5.0.14. Tabla Proyecto

Esta es una de las tablas que tiene relación con otras tablas. Un proyecto está asociado a un usuario que desempeña el rol de Gestor de Proyecto. Por otra parte, también tiene una relación uno a muchos con usuario que son los que participan en el proyecto.

Tomando en consideración lo anterior, se han definido las siguientes restricciones.

- *idProyecto*. Es la llave primaria de la tabla *Proyecto*.

```
PRIMARY KEY (‘idProyecto’)
```

- *Usuario_CURP*. Hace referencia la columna *CURP* de la tabla Usuario y al actualizarse la PK de usuario debe de actualizarse también en Proyecto.

```
CONSTRAINT ‘Proyecto_fk_curp’ FOREIGN KEY (‘Usuario_CURP’)
REFERENCES ‘Usuario’ (‘CURP’) ON UPDATE CASCADE
```

Para la relación Proyecto-Participante se ha creado la tabla *ParticipanteProyecto* que tiene dos columnas, con las siguientes restricciones:

- *Usuario_CURP,Proyecto_idProyecto*. Conforman la llaver primaria.
- *Usuario_CURP*. Llave foránea que hace referencia a la columna *CURP* de la tabla *Usuario*. Si hay alguna actualización de la CURP debe actualizarse también en *ParticipanteProyecto*.

  ```
  CONSTRAINT ‘ParticipanteProyecto_fk_curp’ FOREIGN KEY (‘Usuario_CURP’)
  REFERENCES ‘Usuario’ (‘CURP’) ON UPDATE CASCADE
  ```

- *Proyecto_idProyecto*. Llave foránea que hace referencia a la columna *idProyecto* de la tabla *Proyecto*. Si se elimina un proyecto debe de eliminarse los registros relacionados en *ParticipanteProyecto*.

  ```
  CONSTRAINT ‘ParticipanteProyecto_fk_idProyecto’
  FOREIGN KEY (‘Proyecto_idProyecto’)
  REFERENCES ‘Proyecto’ (‘idProyecto’) ON DELETE CASCADE
  ```

5.0.15. Tabla Iteracion

Esta tabla tiene las siguientes restricciones:

- *numIteracion, Proyecto_idProyecto*. Conforman la llave primaria y son de tipo Integer.

  ```
  PRIMARY KEY (‘numIteracion’,‘Proyecto_idProyecto’)
  ```

- *Proyecto_idProyecto*. Llave foránea que referencia a la columna *idProyecto* de la tabla *Proyecto*. Al eliminarse un proyecto se eliminan sus iteraciones.

  ```
  CONSTRAINT ‘Iteracion_fk_idProyecto’
  FOREIGN KEY (‘Proyecto_idProyecto’)
  REFERENCES ‘Proyecto’ (‘idProyecto’) ON DELETE CASCADE
  ```

- *Fase_idFase*. Llave foránea que referencia a la columna *idFase* de la tabla *Fase*.

  ```
  CONSTRAINT ‘fk_fase’ FOREIGN KEY (‘Fase_idFase’) REFERENCES ‘Fase’ (‘idFase’)
  ```

5.0.16. Tabla Actividad

Hemos definido las siguientes restricciones:

- *idActividad*. Llave primaria.

  ```
  PRIMARY KEY (‘idActividad’)
  ```

- *Disciplina_idDisciplina*. Llave foránea que referencia a la columna *idDisciplina* de la tabla *Disciplina*. Si se actualiza *idDisciplina* en la tabla *Disiciplina* debe de actualizarse también en *Actividad*.

  ```
  CONSTRAINT ‘Actividad_fk_idDisciplina’
  FOREIGN KEY (‘Disciplina_idDisciplina’)
  REFERENCES ‘Disciplina’ (‘idDisciplina’) ON UPDATE CASCADE
  ```

- *Rol_nombreRol*. Llave foránea que referencia a la columna *nombreRol* de la tabla *Rol*. La actualización debe ser en cascada.

  ```
  CONSTRAINT ‘Actividad_fk_nombreRol‘ FOREIGN KEY (‘Rol_nombreRol’)
  REFERENCES ‘Rol’ (‘nombreRol’) ON UPDATE CASCADE
  ```

Por otra parte, una actividad para efectuarse requiere que ciertos artefactos ya se hayan creado, para ello hemos definido la relación *ArtefactoRequeridoActividad* en una tabla que tiene el mismo nombre, con las restricciones:

La llave primaria se define como sigue:

```
PRIMARY KEY (‘Actividad_idActividad‘,‘Artefacto_nombre’)
```

También se añaden las restricciones:

```
CONSTRAINT ‘ArtefactoRequeridoActividad_fk1’
FOREIGN KEY (‘Artefacto_nombre’)
REFERENCES ‘Artefacto’ (‘nombre’) ON UPDATE CASCADE,
CONSTRAINT ‘ArtefactoRequeridoActividad_fk2’
FOREIGN KEY (‘Actividad_idActividad’)
REFERENCES ‘Actividad’ (‘idActividad’) ON UPDATE CASCADE
```

Además una actividad al realizarse produce ciertos artefactos, relación que hemos establecido con la tabla *ArtefactoProducidoActividad*, y tiene las restricciones:

La llave primaria:

```
PRIMARY KEY (‘Actividad_idActividad’,‘Artefacto_nombre’)
```

Restricciones:

```
CONSTRAINT ‘ArtefactoProducidoActividad_fk1’
FOREIGN KEY (‘Artefacto_nombre’)
REFERENCES ‘Artefacto’ (‘nombre’) ON UPDATE CASCADE,
CONSTRAINT ‘ArtefactoProducidoActividad_fk2’
FOREIGN KEY (‘Actividad_idActividad’)
REFERENCES ‘Actividad’ (‘idActividad’) ON UPDATE CASCADE
```

5.0.17. Tabla Tarea

La tabla tiene las siguientes restricciones:

- *numTarea, Iteracion_Proyecto_idProyecto, Iteracion_numIteracion.* Llave primaria de la tabla.

  ```
  PRIMARY KEY (‘numTarea’,‘Iteracion_Proyecto_idProyecto’,
       ‘Iteracion_numIteracion’)
  ```

- *Actividad_idActividad.* Llave foránea que referencia a la columna *Actividad_idActividad* de la tabla *Actividad.*

  ```
  CONSTRAINT ‘Tarea_fk_idActividad’ FOREIGN KEY (‘Actividad_idActividad’)
  REFERENCES ‘Actividad’ (‘idActividad’),
  ```

- *Iteracion_numIteracion,Iteracion_Proyecto_idProyecto.* Llave foránea que referencia a las columnas *numIteracion,Iteracion_Proyecto_idProyecto* de la tabla *Iteracion.*

  ```
  CONSTRAINT ‘Tarea_fk_numIteracion’
  FOREIGN KEY (‘Iteracion_numIteracion’, ‘Iteracion_Proyecto_idProyecto’)
  REFERENCES ‘Iteracion‘ (‘numIteracion’,‘Proyecto_idProyecto’)
  ON DELETE CASCADE ON UPDATE CASCADE
  ```

Tabla TareaUsuario

Hay una relación muchos a muchos entre Tarea-Usuario. Para ello se ha creado la tabla *TareaUsuario* que tiene las siguientes columnas con las restricciones que mencionamos:

- *Usuario_CURP,Tarea_numTarea,Tarea_Iteracion_numIteracion ,Tarea_Iteracion_Proyecto_idProyecto).* Conforman la llave primaria.

```
PRIMARY KEY (‘Usuario_CURP’,‘Tarea_numTarea’,
     ‘Tarea_Iteracion_numIteracion’,
     ‘Tarea_Iteracion_Proyecto_idProyecto’)
```

- *Tarea_numTarea, Tarea_Iteracion_Proyecto_idProyecto, Tarea_Iteracion_numIteracion* es la llave foránea que referencia a la tabla Tarea.

```
CONSTRAINT ‘TareaUsuario_fk_idTarea’
FOREIGN KEY (‘Tarea_numTarea’, ‘Tarea_Iteracion_Proyecto_idProyecto’,
     ‘Tarea_Iteracion_numIteracion’)
REFERENCES ‘Tarea’ (‘numTarea’, ‘Iteracion_Proyecto_idProyecto’,
    ‘Iteracion_numIteracion’)
ON DELETE CASCADE ON UPDATE CASCADE
```

- *Usuario_CURP*. Llave foránea que referencias a un Usuario.

```
CONSTRAINT ‘TareaUsuario_fk_idUsuario’ FOREIGN KEY (‘Usuario_CURP’)
REFERENCES ‘Usuario’ (‘CURP’) ON UPDATE CASCADE
```

Tabla TareaPredecesor

Una tarea no puede iniciarse si no han finalizado ciertas tareas, es decir una tarea es predecesora de otra. Para esta relación se ha creado la tabla *TareaPredecesor* con las siguientes columnas y restricciones.

- La llave primaria está definida por:

```
PRIMARY KEY (‘numTarea’,‘numIteracion’,‘numProyecto’,
     ‘numIteracionP’,‘numProyectoP’,‘numTareaPredecesor’)
```

- La llave primaria de Tarea es foránea en esta tabla, tanto para la tarea y su predecesora.

```
CONSTRAINT ‘fk_tarea’ FOREIGN KEY (‘numTarea’, ‘numIteracion’, ‘numProyecto’)
REFERENCES ‘Tarea’ (‘numTarea‘, ‘Iteracion_Proyecto_idProyecto’,
    ‘Iteracion_numIteracion’)
ON DELETE CASCADE ON UPDATE CASCADE
CONSTRAINT ‘fk_tarea_predecesora’
FOREIGN KEY (‘numTareaPredecesor’,
       ‘numProyectoP’, ‘numIteracionP’)
REFERENCES ‘Tarea’ (‘numTarea’, ‘Iteracion_Proyecto_idProyecto’,
    ‘Iteracion_numIteracion’)
ON DELETE CASCADE ON UPDATE CASCADE
```

5.0.18. Tabla Artefacto

Esta tabla utiliza como llave primaria el atributo nombre de la entidad Artefacto. La cual hemos definido de la siguiente forma:

```
PRIMARY KEY (‘nombre’)
```

Además cuenta con una llave foránea que referencia a la tabla *Rol*, definida así:

```
CONSTRAINT ‘fk_rol’ FOREIGN KEY (‘Rol_nombreRol’)
REFERENCES ‘Rol’ (‘nombreRol’) ON UPDATE CASCADE
```

5.0.19. Tabla Ejemplar

Al crearse una instancia de un Artefacto, le hemos nominado Ejemplar. La tabla correspodiente tiene el mismo nombre. Como llave primaria utiliza el número del proyecto al cual pertenece y el nombre del artefacto, por lo que se ha definido como sigue:

```
PRIMARY KEY (‘Artefacto_nombre’,‘idEjemplar’,‘Proyecto_idProyecto’)
```

Por otra parte, *idEjemplar* y *Proyecto_idProyecto* son llaves foráneas que referencian a *Ejemplar* y *Proyecto* respectivamente. Por lo que se definen las siguientes restricciones:

```
CONSTRAINT ‘Ejemplar_ibfk_1’ FOREIGN KEY (‘Artefacto_nombre’)
REFERENCES ‘Artefacto‘ (‘nombre’)
ON DELETE NO ACTION ON UPDATE NO ACTION,
CONSTRAINT ‘Ejemplar_ibfk_2’ FOREIGN KEY (‘Proyecto_idProyecto’)
REFERENCES ‘Proyecto’ (‘idProyecto’)
ON DELETE NO ACTION ON UPDATE NO ACTION
```

5.0.20. Tabla Plantilla

Un *Artefacto* puede tener una plantilla que se utilizará para crear un ejemplar del artefacto. Se creó la tabla *Plantilla* con las siguientes restricciones:

Como llave primaria utiliza el número del proyecto al cual pertenece y el nombre del artefacto, por lo que se ha definido como sigue:

```
PRIMARY KEY (‘Artefacto_nombre’,‘idEjemplar’,‘Proyecto_idProyecto’)
```

Por otra parte, *idEjemplar* y *Proyecto_idProyecto* son llaves foráneas que referencian a *Ejemplar* y *Proyecto* respectivamente. Por lo que se definen las siguientes restricciones:

```
CONSTRAINT ‘Plantilla_ibfk_1’ FOREIGN KEY (‘Artefacto_nombre’)
REFERENCES ‘Artefacto‘ (‘nombre’)
ON DELETE NO ACTION ON UPDATE NO ACTION,
CONSTRAINT ‘Plantilla_ibfk_2’ FOREIGN KEY (‘Proyecto_idProyecto’)
REFERENCES ‘Proyecto’ (‘idProyecto’)
ON DELETE NO ACTION ON UPDATE NO ACTION
```

Las restricciones de integridad referencial definidas en este capítulo e implementadas en la base de datos nos permitirán un mejor manejo de las actualizaciones, borrados e inserciones en la base de datos.

Capítulo 6

Hibernate y clases persistentes

6.1. Configuración del Ambiente de Desarrollo

Antes de empezar con la implementación primeramente tenemos que configurar el ambiente de desarrollo. Así tenemos lo siguiente a realizar:

- Tener instalado Eclipse Java EE IDE for Web Developers.
- Tener servidor Tomcat 6.1
- Crear un Proyecto Web Dinámico.
- Configurar el proyecto para el desarrollo con Struts.
- Configurar el proyecto para trabajar con Hibernate.
- Desarrollar

6.1.1. Elclipse

Como primer paso tenemos que descargar el IDE que usaremos durante el desarrollo de la aplicación. Para ello se descargó el Eclipse Java EE IDE for Web Developers, de la página oficial *http://www.eclipse.org/*, esta versión es la que nos será util ya que desarrollaremos una aplicación Web.

6.1.2. Tomcat

Apache Tomcat es un un contenedor de servlets el cual implementa las especificaciones de los servlets y de JavaServer Pages (JSP). Lo descargamos de la página oficial *http://tomcat.apache.org/*.

6.1.3. Creación de un Proyecto Web

Una vez descargado el IDE, crearemos un nuevo proyecto Web. Para ello nos dirigimos al menú File –> Other –> Dynamic Web Project. Proporcionamos el nombre de la aplicación, y tendremos creado el nuevo proyecto. Posteriormete tenemos que especificar el servidor web que utilizaremos para las pruebas. Para ello click derecho en la opción de *Servers* del IDE, despues nos vamos a *New –> Server* , tras lo cual especificamos la versión del Servidor Tomcat descargado, y especificamos la ruta de los archivos descargados. Con esto ya tendremos el servidor para nuestra aplicación web.

6.1.4. Configuración del Proyecto y Struts

Ahora tenemos que configurar el nuevo proyecto creado para poder desarrollar utilizando Struts. Para ello tenemos que descargar la implementación de Struts desde la página oficial, *http://struts.apache.org/*, en este caso desarrollaremos con la versión 2.1. Después tenemos que copiar los archivos:

1. commons-fileupload-X.X.X.jar
2. commons-io-X.X.X.jar
3. commons-logging-X.X.X.jar
4. commons-logging-api.X.X.jar
5. freemarker-X.X.X.jar
6. ognl-X.X.X.jar
7. struts2-core-X.X.X.X.jar
8. xwork-core-X.X.X.jar

Estos archivos contienen las clases necesarias para el desarrollo básico con Struts. Estos archivos los copiaremos a la carpeta WEB-INF\lib. Posteriormente conforme vayamos necesitando otras clases de Struts las iremos agregando.

```
<filter>
  <filter-name>struts2</filter-name>
  <filter-class>org.apache.struts2.dispatcher.ng.filter.
      StrutsPrepareAndExecuteFilter</filter-class>
</filter>

<filter-mapping>
  <filter-name>struts2</filter-name>
   <url-pattern>/*</url-pattern>
</filter-mapping>
```

Listing 6.1: web.xml

Una vez realizado lo anterior tenemos que habilitar struts. Para ello agregamos en el archivo *web.xml* el código mostrado en 6.1:

A continuación tenemos que crear un archivo llamado *struts.xml*. Este archivo nos servirá para realizar los mapeos de un evento a una clase, es decir, en el especificaremos qué clase responderá a cierto evento generado. Este archivo lo pondremos en la carpeta *src*. Este archivo tiene una estructura como la mostrada en 6.2:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
    "http://struts.apache.org/dtds/struts-2.0.dtd">

<struts>
  <package name="rol" extends="struts-default">
    <result-types>
      <result-type name="tiles" class="org.apache.struts2.views.tiles.
          TilesResult" />
    </result-types>

    <action name="rolesSistema" class="com.gestion.vista.rol.
        RolSistemaAction"
      method="rolesSistema">
      <result name="success" type="tiles">RolesSistema</result>
      <result name="error" type="tiles">Eror</result>
    </action>
     ................
  </package>
</struts>
```

Listing 6.2: struts.xml

Con las palabras *package name="rol"*, creamos una agrupación de acciones bajo el nombre de *rol* y que tendrá las mismas caracteristicas de *"struts-default"*. Por otra

parte las palabras *result-type name="tiles"*, estamos definiendo un resultado que identificaremos con el ombre de *"tiles"*, y con la palabra *"class"* definimos que el resultado bajo el nombre de *"tiles"* será de la clase *org.apache.struts2.views.tiles.TilesResult*.

Los Tiles son una especie de plantillas. Generalmente son usados para generar componentes de vistas que pueden reutilizarse. En un Tile se pueden definir las partes de la vista tales como el header, footer, body entre otros y después reutilizar ese Tile para definir otra vista con solo cambiar el valor de una de las variables, por ejemplo el body. A continuación mostramos un ejemplo (ver el código 6.3) de la definición de un Tile, y después explicaremos la configuración necesaria para poder trabajar con Tiles.

```
<?xml version="1.0" encoding="UTF-8" ?>

<!DOCTYPE tiles-definitions PUBLIC
       "-//Apache Software Foundation//DTD Tiles Configuration 2.0//EN"
       "http://tiles.apache.org/dtds/tiles-config_2_0.dtd">

<tiles-definitions>
  <!-- Estructura general -->

  <definition name="basic2.layout" template="/jsp/plantilla/
      estructuraGeneral.jsp">
    <put-attribute name="pageTitle" value="Inicio" />
    <put-attribute name="header" value="/jsp/plantilla/header.jsp"></put
        -attribute>
    <put-attribute name="footer" value="/jsp/plantilla/footer.jsp"></put
        -attribute>
    <put-attribute name="middle" value="/jsp/plantilla/middle.jsp"></put
        -attribute>
  </definition>

  <!-- ########################## -->
  <!-- Caso de Prueba con basic2.layout -->
  <!-- ########################## -->
  <definition name="PruebaEstilo" extends="basic2.layout">
    <put-attribute name="middle" value="/jsp/personal/centroPrueba.jsp">
        </put-attribute>
  </definition>
   ......
</tiles-definitions>
```

Listing 6.3: tile.xml

En este caso definimos un tile o plantilla llamado *"basic2.layout"*, especificamos un atributo *pageTitle* cuyo valor será el título de la página. Definimos un *header, footer, middle* los cuales estás asociados a una página JSP. Posteriormente si queremos crear una vista que es similar a esta vista, basta con cambiar el valor asociado a un atributo. En el ejemplo mostrado, creamos otra vista cambiando sólo el valor del

atributo *middle*, con eso ya tendremos otra vista sin tener que repetir todo el código nuevamente.

Habrá observado que para el tile *basic2.layout* tiene un atributo *template="/jsp/plantilla/estructuraGeneral.jsp"*. Ese archivo define la estructura de la vista, y luce com se ve 6.4:

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8" %>
<%@ taglib prefix="s" uri="/struts-tags" %>
<%@ taglib uri="http://tiles.apache.org/tags-tiles" prefix="tiles" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://
    www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title><tiles:getAsString name="pageTitle"/></title>
    <link href="<s:url value="/jsp/css/admin.css"/>" rel="stylesheet"
          type="text/css" />
<link type="text/css" href="/PrototipoWeb/jsp/css/jquery-ui-1.8.2.custom
    .css" rel="Stylesheet" />
</head>
<body>

<div id="main">
  <!-- header -->
  <tiles:insertAttribute name="header" />
  <!-- middle -->
  <tiles:insertAttribute name="middle" />

  <!-- footer -->
  <tiles:insertAttribute name="footer" />
</div>

</body>
</html>
```

Listing 6.4: estructuraGeneral.jsp

Y esta será la estructura para toda vista que creemos en nuestra aplicación.

Ahora bien, para poder hacer uso de los Tiles tenemos que realizar lo siguiente. Como primer paso tenemos que especificar el archivo que contendrá los tiles, para ello en el archivo *web.xml*, agregamos las lineas mostradas en 6.5.

Además de lo anterior, tenemos que descargar los archivos:

1. struts2.tiles-plugin

2. tiles-api
3. tiles-compat
4. tiles-core
5. tiles-jsp
6. tiles-portlet
7. tiles-servlet

```
<context-param>
  <param-name>org.apache.tiles.impl.BasicTilesContainer.
      DEFINITIONS_CONFIG</param-name>
  <param-value>/WEB-INF/tiles.xml</param-value>
</context-param>

<listener>
  <listener-class>org.apache.struts2.tiles.StrutsTilesListener</
      listener-class>
</listener>
```

Listing 6.5: web.xml(2)

Estos archivos deberán ir en la carpeta *WEB-INF\lib*, con esto tendremos la configuración necesaria para poder empezar a trabajar con Tiles.

6.1.5. Configuración de Hibernate

Se tiene que crear un archivo llamado *"hibernate.cfg.xml"*, el cual tendrá la configuración para Hibernate, de acuerdo a la base de datos a utlizar y el lenguaje de consultas. A continuación mostramos nuestro archivo de configuración, el cual es bastante explicativo.

```
<?xml version='1.0' encoding='utf-8'?>
<!DOCTYPE hibernate-configuration PUBLIC
        "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
        "http://hibernate.sourceforge.net/hibernate-configuration-3.0.
            dtd">

<hibernate-configuration>

  <session-factory>

    <!-- Driver -->
    <property name="connection.driver_class">
      com.mysql.jdbc.Driver
    </property>
    <!-- URL de la base de datos -->
    <property name="connection.url">
      jdbc:mysql://localhost/PATIN
    </property>
    <!-- Nombre de usuario y contrasena a utilizar para la conexion -->
    <property name="connection.username">usuario</property>
    <property name="connection.password">password</property>

    <!-- JDBC connection pool -->
    <property name="connection.pool_size">1</property>

    <!-- dialecto SQL -->
    <property name="dialect">
      org.hibernate.dialect.MySQLDialect
    </property>

    <!-- Enable Hibernate's automatic session context management -->
    <property name="current_session_context_class">thread</property>

    <!-- Disable the second-level cache  -->
    <property name="cache.provider_class">
      org.hibernate.cache.NoCacheProvider
    </property>

    <!-- Mostrar o no las sentencias sql ejecutadas por hibernate -->
    <property name="show_sql">true</property>

    <property name="hbm2ddl.auto">update</property>

```

```
    <!-- clases mapeadas -->
44
  </session-factory>
46
</hibernate-configuration>
```

Listing 6.6: hibernate.cfg.xml

Este archivo de configuración lo colocaremos en la carpeta *src*, puesto que es allí donde lo buscará Hibernate. Ahora tendremos que descargar los siguientes archivos:

1. antlr
2. commons-collections
3. commons-lang
4. commons-logging
5. dom4j
6. ejb3-persistance
7. hibernate-annontations
8. hibernate-commons-anotation
9. hibernate-validator
10. hibernate3
11. javassist
12. jta
13. junit
14. mysql-connector-java (porque vamos a trabajar con mysql)
15. slf4j
16. slf4j-log4j12
17. struts2-fullhibernate-plugin

Con estos pasos, tenemos la configuración necesaria para trabajar con Hibernate. De acuerdo a la documentación de Hibernate, para conocer que clases están mapeadas busca los archivos *.hbm.xml* y después busca clases que estén mapeadas con anotaciones. Debido a que nosotros trabajaremos con anotaciones y no tendremos ninguna clase mapeada con archivos *hbml.xml*, no tendremos que modificar este comportamiento, de lo contrario sí será necesario.

6.2. Clases Persistentes

Las clases presistentes son clases en una aplicación que implementan las entidades del problema empresarial. Para la implementación de las clases persistentes, se implementarán considerando lo siguiente, de acuerdo a la documentación de Hibernate.

- Para especificar que una clase es persistente se debe de utilizar la palabra *@Entity*
- Para definir a que tabla será mapeada la clase se utiliza *@Table(name="nombreTabla")*.
- Para indicar el identificador de la clase, que corresponde finalmente con la llave primaria de la tabla asociada, se utiliza la palabra *@Id*.
- Para declarar identificadores compuestos se usa *@EmbeddedId*.
- Para informar a Hibernate a qué columna se mapeará un atributo se utiliza *@Column(name="nombreColumna")*. Además pueden incluirse los siguientes atributos: *unique*, agrega la restricción de que el valor de la columna no se puede repetir en la tabla; *nullable*, true si el valor del campo puede ser null, false en caso contrario; *insertable*, si la columna puede formar parte de una sentencia de inserción; *updatable*, si la columna puede formar parte de una sentencia de actualización.
- Lost tipos *java.sql.Clob, Character[], char[] y java.lang.String* son almacenados como un tipo *Clob* utilizando el elemento *@Lob*. Por su parte los tipos de datos *java.sql.Blob*, *Byte[]*, *byte[]* y tipos serializables seran almacendos como *Blob*.
- Para las relaciones uno a uno se utiliza *@OneToOne*.
- Para la asociacion uno a muchos se utiliza *@ManyToOne*.
- Para las relaciones muchos a muchos se utiliza *@ManyToMany*.
- Para hacer el mapeo de objetos tipo java.Util.Date se utiliza alguno de: *@Temporal(TemporalType.DATE — TIME — TIMESTAMP)*

A continuación algunos estándares útiles.

- Si no se especifica el nombre de la tabla a la que se mapeará la clase, Hibernate asume que la tabla tiene el mismo nombre que la clase.
- Todos los atributos de una clase se consideran persistentes, a menos que se especifique lo contrario utilizando *@Transient*

Comenzaremos con la implementación de una de las clases más sencillas, la clase Rol, mostrado en 6.7.

6.2.1. Rol.java

```
1  package com.gestion.modelo.rol;

3  import java.io.Serializable;
   import javax.persistence.Column;
5  import javax.persistence.Entity;
   import javax.persistence.Id;
7
   @Entity
9  public class Rol implements Serializable{
   private String nombreRol;
11 private String descripcion;
   private String urlDescripcion;
13 private String categoria;
   private String habilidades;
15 private String criterioAsignacion;

17 public Rol(){}

19 @Id
   public String getNombreRol() {
21    return nombreRol;
   }
23
   /**
25   *  métodos getter y setter
     */
27 }
```

Listing 6.7: Rol.java

Toda clase persistente debe implementar la interfaz *Serializable*. Además debe contar con un constructor sin argumentos, el cual es utilizado por Hibernate para crear una instancia de la clase. En este caso y en la mayoría de nuestras clases persistentes no será necesario especificar a qué columna se mapeará cada atributo, debido a que el nombre de los atributos son iguales al nombre de columnas en la base de datos. Esto lo hemos hecho por conveniencia. Sin embargo, en caso de que el nombre del atributo sea distinto al nombre de la columna en la BD, tendremos que utilizar *@Column* para realizar el mapeado correcto.

6.2.2. Disciplina.java

Otra de las entidades de nuestra Base de Datos es la Disciplina, la cual implementamos como se observa en 6.8.

```
1  package com.gestion.modelo.proyecto;

3  import java.io.Serializable;
   import javax.persistence.Entity;
5  import javax.persistence.Id;

7  @Entity
   public class Disciplina implements Serializable {
9    private Integer idDisciplina;
     private String nombre;
11   private String descripcion;

13   public Disciplina(){}

15   @Id
     public Integer getIdDisciplina() {
17     return idDisciplina;
     }
19   /**
       * métodos getter y setter
21     */
   }
```

Listing 6.8: Disciplina.java

6.2.3. Fase.java

Ahora implementaremos la entidad Fase mediante la clase Fase.java mostrado 6.9.

```
package com.gestion.modelo.proyecto;

import java.io.Serializable;
import javax.persistence.Entity;
import javax.persistence.Id;

@Entity
public class Fase implements Serializable{
  private long idFase;
  private String nombre;
  private String descripcion;

  public Fase(){}

  @Id
  public long getIdFase() {
    return idFase;
  }
  /**
    * métodos getter y setter
    */
}
```

Listing 6.9: Fase.java

Hasta ahora hemos realizado el mapeo de clases que no tienen ninguna relación con otras clases, lo cual evidentemente fue sencillo, una vez que se conocen las reglas de mapeado. Ahora realizaremos el mapeo de otras clases que sí tienen relación con otras clases, e iremos explicando la forma de hacerlo.

6.2.4. Proyecto.java

La entidad Proyecto es una de las más importantes en nuestra Base de Datos. Por el momento mostraremos una implementación parcial, a fin de evitar confusiones. El código parcial de la clase se observa en 6.10.

Como se puede apreciar, en este caso nuestra clase Proyecto tiene una relación uno a muchos con la clase Usuario, también una relación uno a uno con Usuario, y una relación uno a muchos con la clase Iteracion. Llegados a este punto, explicaremos a detalle el mapeo de relaciones con Hibernate.

```
package com.gestion.modelo.proyecto;

import java.io.Serializable;
import java.sql.Date;
import java.util.ArrayList;
import java.util.HashSet;
import java.util.List;
import java.util.Set;
import javax.persistence.*;
import com.gestion.modelo.usuario.Usuario;

/*
 * Esta clase contiene la informacion relacionada con un proyecto.
 */
@Entity
public class Proyecto implements Serializable{
  private long idProyecto;
  private String nombre;
  private String descripcion;
  private String cliente;
  private Date fechaInicio;
  private Date fechaFin;
  private Set<Usuario> listaUsuarios = new HashSet<Usuario>();
  private Usuario jefe;
  private List<Iteracion> iteraciones = new ArrayList<Iteracion>();

  @Id
  public long getIdProyecto() {
    return idProyecto;
  }
```

Listing 6.10: Proyecto.java

Relación Uno a Uno

La relación uno a uno se presenta cuando una entidad A sólo puede estar relacionado con una entidad B. A nivel de base de datos es cuando un registro de una tabla X, sólo puede estar relacionado con un único registro de la tabla Y. Por ejemplo, un Proyecto sólo puede tener un único gestor de proyecto, por lo que un registro en la tabla Proyecto estará relacionado con un único registro en la tabla Usuario. La implementación de dicha relación se puede hacer de diversas maneras. Una de las formas es que la tabla Proyecto tenga como llave foránea la llave primaria de Usuario, agregando como restricción que dicha llave foránea debe ser única en la tabla Proyecto. Otra forma de hacerlo es que tanto el registro de Proyecto y Usuario compartan una misma llave, es decir, la llave primaria para Usuario será la misma para Proyecto. Una última forma de implementarlo es utilizando una tabla adicional en donde se almacenará la llave primaria de Proyecto y la de Usuario en un registro, teniendo como restricción que dichos valores deben ser únicos en la tabla. Dependiendo de la opción elegida el mapeo con Hibernate variará.

Para la situación que estamos tratando, la relación entre un Proyecto y el Usuario que es jefe de dicho proyecto, en la tabla Proyecto se almacena la llave primaria de dicho Usuario. Por lo que la explicación dada a continuación será para este caso.

Como primer aspecto a tener en cuenta, una relación Uno a Uno se implementa lógicamente usando *@OneToOne*, ésta anotación tiene 5 argumentos opcionales que son:

- *cascade*. Permite especificar cómo se afectará a la entidad con la que se tiene la relación. Puede tomar uno de los siguientes valores del tipo *Enum<CascadeType>* que son:
  - ALL: Indica que todas las operaciones serán en cascada.
  - MERGE: Indica que las operaciones de tipo MERGE se efectuarán en cascada.
  - PERSIST:
  - REFRESH: Indica que las operaciones de actualización se harán en cascada.
  - REMOVE: Indica que las operaciones de borrado se efectuarán en cascada.
- *fetch*. Permite especificar de qúe forma será cargada la entidad con la que se tiene la relación. Toma uno de los valores de *Enum<FetchType>*:
  - EAGER: Indica que los datos del registro con la que se tiene la relación se recuperará inmediatamente.
  - LAZY: Indica que los datos del registro con la que se tiene la relación se recuperará sólo cuando se necesite.

- *mappedBy*. Permite especificar el nombre del atributo propietario de la relación. Es de tipo *String*.
- *optional*. Es de tipo *boolean*. Si es true indica que la relación es opcional, obligatoria en caso contrario.
- *targetEntity*. Es de tipo *Class*. Indica la clase con la que se tiene la relación.

Otra de las anotaciones usadas para el mapeado de las relaciones es *@JoinColumn*, la cual tiene 8 argumentos opcionales, y son:

- *columnDefinition*.
- *insertable*. Indica si la columna puede ser incluida en una sentencia insert. Dafault *true*.
- *name*. El nombre de la columna que es llave foránea. Es de tipo *String*.
- *nullable*. Si es *true*, el valor de la llave foránea puede ser *null*. Dafult *true*.
- *referencedColumnName*. El nombre de la columna de la donde se almacena el valor de la llave foránea. Si no se especifica, se asume que es el nombre de la columna de la llave primaria de la tabla a la cual está mapeada la entidad destino de la relación.
- *table*. El nombre de la tabla en donde está la columna. Si no se especifica una tabla se asume que es la tabla a la cual está mapeada la entidad destino de la relación.
- *unique*. Si es *true*, es llave única en la tabla. Dafault *false*
- *updatable*. Si es *true*, la columna puede incluirse en una sentencia de actualización. Default *true*.

Con estas nociones procederemos a mapear nuestra clase Proyecto y su relación con Usuario, la cual queda como se ve en 6.11.

Como no se especificó ningun valor para *cascade, fetch, y mappedBy*, estos tomarán los valores predeterminados que son *ninguna operación en cascada, EAGER y " "* respectivamente, lo cual es correcto para la relación.

Al no haberse especificado *insertable, nullable, referencedColumnName, table, unique y updatable* , estas tomarán los valores predeterminados que serían *true,true, "Usuario_CURP", "Usuario",false y true*. Estos valores son apropiados para la relación, y hemos evitado especificarlos gracias a que hay una consistencia tanto en los nombres de las tablas con los nombres de la clases, y en los nombres de las columnas de las tablas con los nombres de los atributos de las clases.

```
@OneToOne
@JoinColumn(name="Usuario_CURP")
public Usuario getJefe() {
  return jefe;
}
public void setJefe(Usuario jefe) {
  this.jefe = jefe;
}
```

Listing 6.11: Proyecto.java

Relación Uno a Muchos

Otra de la relaciones comunes es la de Uno a Muchos. En esta relación un registro de una tabla principal X, puede estar relacionado con varios registros de una tabla secundaria Y. Pero un registro de la tabla Y sólo puede estar relacionado un único registro de la tabla X. Ejemplo de esta clase de relación es la de un Padre-Hijo. Un Padre (tabla principal) tiene varios Hijos(tabla secundaria), pero un Hijo sólo tiene un Padre. La forma de implementar este tipo de relación en la base de datos es que en la tabla secundaria se almacene una llave foránea del registro de la tabla principal. Para nuestro caso particular, la relación entre Proyecto e Iteraciones es de uno a muchos. La razón es que un proyecto tiene muchas iteraciones, pero una iteración está asociada a un único proyecto.

Para realizar un mapeo de esta clase de relación se utiliza *@OneToMany*, el cual tiene los mismos argumentos que el *@OneToMany*. Se utiliza también el *@JoinColumn*. En 6.12 mostramos el mapeado.

Recordando lo que habiamos mencionado, la anotación *mappedBy*, se utiliza para especificar el nombre del atributo mapeado en la entidad propietaria de la relación, esto es necesario cuando se desea que la relación sea bidireccional. Con el mapeado estamos indicando que el lado que hace el mapeo es Iteracion en el atributo *proyecto*. En 6.13 se muestra el mapeo:

A fin de comprender mejor este mapeo, en las figuras 6.1 y 6.2 mostramos las tablas de dichas entidades.

Además mostramos parcialmente la clase Iteracion en 6.14.

La tabla Iteracion es la que tiene una llave foránea de Proyecto, por lo que es la propietaria de la relación, razón por la cual el mapeado se hace en la clase Iteración y no en la clase Proyecto. De acuerdo a la documentación *name* es el nombre de la columna que es llave foránea, y efectivamente *Proyecto_idProyecto* es la llave foránea

```
@OneToMany(mappedBy="proyecto")
public List<Iteracion> getIteraciones() {
  return iteraciones;
}
```

Listing 6.12: Mapeado Proyecto-Iteracion

```
/**
 * Varias entidades Iteracion estas asociadas con un proyecto, para
    saber cual es
 * basta consultar la columna 'Proyecto_idProyecto'
 * Al recuperar la iteracion recupera el Proyecto al que pertenece
 * @return
 */
@ManyToOne
@JoinColumn(name="Proyecto_idProyecto",insertable=false, updatable=
    false)
public Proyecto getProyecto() {
  return proyecto;
}
public void setProyecto(Proyecto proyecto) {
  this.proyecto = proyecto;
}
```

Listing 6.13: Mapeado Iteracion-Proyecto

en la tabla Iteracion. Los valores de *insertable=false, updatable=false* son false debido a que no podemos actualizar dicha llave foránea ni insertar un registro de un Proyecto al insertar una iteración. El proceso es exactamente al revés, primero se inserta un registro de proyecto y posteriormente se insertan tantas iteraciones como se desee.

Figura 6.1: Tabla Iteracion

Figura 6.2: Tabla Proyecto

```
@Entity
public class Iteracion implements Serializable{
  private IdIteracion id;
  private String nombre;
  private String descripcion;
  private Date fechaInicio;
  private long indice;
  private Fase fase;
  private Proyecto proyecto;
  ......
}
```

Listing 6.14: Iteracion.java

Relación Muchos a Muchos

Continuando con el mapeo de la clase Proyecto, surge otro tipo de relación. De acuerdo a los requerimientos en un proyecto pueden participar muchos usuarios y un usuario puede participar simultáneamente en varios proyecto, por lo que es una relación muchos a muchos. La forma de implementarlo en la base de de datos es creando un tabla adicional que nos permita almacenar los pares id proyecto y id usuario. Según se expuso anteriormente para esto se creó la tabla Participante_Proyecto, la cual mostramos en

Figura 6.3: Insertar imagen correcta

Esta tabla tiene dos campos que son *Usuario_CURP,Proyecto_idProyecto* que permiten tal relación. Llegados a este punto es momento de explicar otra de las anotaciones para el mapeado, la anotación *@JoinTable*. Esta es usada cuando utilizamos una tabla intermedia para una relación. Tiene 4 argumentos opciones que son:

- *catalog*. El nombre del catálogo.
- *inverseJoinColumns*. Es del tipo *JoinColumn[]*. Indica las columnas que contienen la llave foránea de la entidad que no es propietaria de la relación.
- *joinColumns*. Es del tipo *JoinColumn[]*. Indica las columnas que contienen la llave foránea de la entidad propietaria de la relación.
- *name*. El nombre de la tabla intermediaria.
- *schema*. El nombre del esquema.
- *uniqueConstraints*. Es del tipo *UniqueConstraint[]*. Restricciones.

Siguiendo estás especificaciones el mapeado queda como se ve en 6.15.

```
1  /**
    * Hay una asociacion Muchos a Muchos entre Proyecto y Usuario
3   * @return
    */
5  @ManyToMany(targetEntity=Usuario.class)
   @JoinTable(name="ParticipanteProyecto",
7        joinColumns=@JoinColumn(name="Proyecto_idProyecto"),
         inverseJoinColumns=@JoinColumn(name="Usuario_CURP"))
9  public Set<Usuario> getListaUsuarios() {
     return listaUsuarios;
11 }
   public void setListaUsuarios(Set<Usuario> listaPersonal) {
13   this.listaUsuarios = listaPersonal;
   }
```

Listing 6.15: Mapeado Proyecto-Usuario

Con el mapeo realizado anteriormente, nuestra clase Proyecto queda de la siguiente forma:

```
@Entity
public class Proyecto implements Serializable{
  private long idProyecto;
  private String nombre;
  private String descripcion;
  private String cliente;
  private Date fechaInicio;
  private Date fechaFin;
  private Set<Usuario> listaUsuarios = new HashSet<Usuario>();
  private Usuario jefe;
  private List<Iteracion> iteraciones = new ArrayList<Iteracion>();

  /*
   * Constructor sin argumentos para Hibernate
   */
  public Proyecto(){

  }
  /*
   * con la anotacion @Id indicamos que este atributo
   * es la clave primaria de la entidad Proyecto
   */
  @Id
  public long getIdProyecto() {
    return idProyecto;
  }
  public void setIdProyecto(long idProyecto) {
    this.idProyecto = idProyecto;
  }
  public String getNombre() {
    return nombre;
  }
  public void setNombre(String nombre) {
    this.nombre = nombre;
  }
  public String getDescripcion() {
    return descripcion;
  }
  public void setDescripcion(String descripcion) {
    this.descripcion = descripcion;
  }
  public String getCliente() {
    return cliente;
  }
  public void setCliente(String cliente) {
    this.cliente = cliente;
  }

```

```
    /**
50   * Hay una asociacion Muchos a Muchos entre Proyecto y Usuario
     * @return
52   */
    @ManyToMany(targetEntity=Usuario.class)
54  @JoinTable(name="ParticipanteProyecto",joinColumns=@JoinColumn(name="
        Proyecto_idProyecto"),
         inverseJoinColumns=@JoinColumn(name="Usuario_CURP"))
56  public Set<Usuario> getListaUsuarios() {
      return listaUsuarios;
58  }
    public void setListaUsuarios(Set<Usuario> listaPersonal) {
60    this.listaUsuarios = listaPersonal;
    }
62
    public Date getFechaInicio() {
64    return fechaInicio;
    }
66  public void setFechaInicio(Date fechaInicio) {
      this.fechaInicio = fechaInicio;
68  }

70  public Date getFechaFin() {
      return fechaFin;
72  }
    public void setFechaFin(Date fechaFin) {
74    this.fechaFin = fechaFin;
    }
76
    /**
78   * Un proyecto tiene solo un jefe de proyecto.
     * con el codigo de abajo se ha establecido al relacion uno  a uno
80   * Recupera tambien al Jefe
     */
82  @OneToOne
    @JoinColumn(name="Usuario_CURP")
84  public Usuario getJefe() {
      return jefe;
86  }
    public void setJefe(Usuario jefe) {
88    this.jefe = jefe;
    }
90
    /**
92   * Un Proyecto esta asociado a varias Iteraciones
     */
94  @OneToMany(mappedBy="proyecto")
    //@Column(name="id")
96  public List<Iteracion> getIteraciones() {
      return iteraciones;
98  }
```

```
100   public void setIteraciones(List<Iteracion> iteraciones) {
        this.iteraciones = iteraciones;
102   }

104   /**
       * Agrega un nuevo personal a lista de personal del proyecto.
106    * @param nuevo Una instancia de Personal que se agregara al proyecto
           actual.
       */
108   public void agregarPersonal(Usuario nuevo){
        this.listaUsuarios.add(nuevo);
110   }

112   public void eliminarPersonal(Usuario p){
        if( this.listaUsuarios.contains(p)){
114       System.out.println("Se eliminara: "+p.getIdUsuario());
        }
116   }

118 }
```

Listing 6.16: Mapeado Proyecto-Usuario

6.2.5. Usuario.java

Otra de las entidades importantes en nuestra base de datos es la entidad Usuario. A continuación mostraremos el mapeado y después explicaremos algunos detalles.

```
package com.gestion.modelo.usuario;

import java.io.Serializable;
import java.util.ArrayList;
import java.util.HashSet;
import java.util.List;
import java.util.Set;

import javax.persistence.*;

import com.gestion.modelo.proyecto.Proyecto;
import com.gestion.modelo.proyecto.Tarea;
import com.gestion.modelo.rol.Rol;

@Entity
public class Usuario implements Serializable{
  private String idUsuario;
  private String nombre;
  private String ape_paterno;
  private String ape_materno;
  private String correo;
  private String telefono;
  private String login;
  private String pass_word;
  private Set<Rol> roles= new HashSet<Rol>();
  private List<Proyecto> proyectos = new ArrayList<Proyecto>();
  private String nombreCompleto;
  private List<Tarea> tareas;


  public Usuario(){

  }

  @Id
  @Column(name="CURP")
  public String getIdUsuario() {
    return idUsuario;
  }
    .........

  @ManyToMany(targetEntity=Rol.class)
  @JoinTable(name="UsuarioRol",joinColumns=@JoinColumn(name="
      Usuario_CURP"),
       inverseJoinColumns = @JoinColumn( name="Rol_nombreRol"))
  public Set<Rol> getRoles() {
```

```
46      return roles;
    }
48
    ........
50
    @Transient
52  public String getNombreCompleto() {
      return this.nombre + " "+ape_paterno+" "+ape_materno;
54  }

56  public void setNombreCompleto(String nombreCompleto) {
      this.nombreCompleto = nombreCompleto;
58  }

60  @ManyToMany(mappedBy="listaUsuarios")
    public List<Proyecto> getProyectos() {
62    return proyectos;
    }
64
    public void setProyectos(List<Proyecto> proyectos) {
66    this.proyectos = proyectos;
    }
68  @ManyToMany(mappedBy="recursos")
    public List<Tarea> getTareas() {
70    return tareas;
    }
72
    public void setTareas(List<Tarea> tareas) {
74    this.tareas = tareas;
    }
76 }
```

Listing 6.17: Usuario.java

En este caso, además de especificar el atributo identificador de la clase, hemos indicado también el nombre de la columna a la cual será mapeado el atributo, esto debido a que el nombre del atributo es *idUsuario* mientras que el nombre del campo en la tabla Usuario es *CURP*.

Por otra parte, el mapeado mostrado en 6.18 indica que: la tabla que representa la relación es *UsuarioRol*, la anotación *joinColumns=@JoinColumn(name="Usuario_CURP")* especifican que la columna *Usuario_CURP* es la llave foránea de la entidad propietaria de la realación, es decir, Usuario; la anotación *inverseJoinColumns = @JoinColumn(name="Rol_nombreRol")* indica la llave foránea de la entidad objetivo de la relación, es decir, la entidad Rol.

Otra de las capacidades que ofrece Hibernate, es que una clase persistente, puede tener atributos que no corresponden con una columna en la Base de Datos. En ocasiones será necesario tener ciertos atributos en la clase que serán útiles para ciertas operaciones pero que no serán mapeadas a la base de datos. Para lograr dicho

```
@ManyToMany(targetEntity=Rol.class)
@JoinTable(name="UsuarioRol",joinColumns=@JoinColumn(name="Usuario_CURP"
    ),
    inverseJoinColumns = @JoinColumn( name="Rol_nombreRol"))
public Set<Rol> getRoles() {
    return roles;
}
```

Listing 6.18: Mapeado Usuario-Rol

comportamiento se utiliza *@Transient*. Para la clase Usuario lo hemos indicado para el atributo *nombreCompleto*, una información necesaria frecuentemente, pero que será ignorada por Hibernate al momento de que realice operaciones en la BD.

En cuanto a la relación Usuario-Proyecto ya la explicamos al momento de mapear el Proyecto.

Ahora comentaremos la relación entre Usuario y Tareas. Es una relación Muchos a Muchos. En este caso hemos especificado que es una relación bidireccional, es decir un usuario puede conocer sus tareas y una entidad tarea puede conocer a que usuarios está asignado. En 6.19 hemos realizado el mapeo.

```
@ManyToMany(mappedBy="recursos")
public List<Tarea> getTareas() {
    return tareas;
}
```

Listing 6.19: Mapeado Usuario-Tarea

En la clase Tareas.java el mapeo es como se ve en 6.20.

Hemos usado la anotación *@JoinColumn*, con algunos aspectos nuevos. Ahora incluimos la anotación *referencedColumnName*. Primero mostraremos las tablas que corresponden a la entidad Usuario, Tarea y la tabla que permite la relación a fin de comprender mejor la razón de esta anotación.

Ver tabla Tarea en figura 6.4.

La tabla Usuario se puede ver en la figura 6.5.

La tabla que permite la relación entre ambas entidades tiene los siguientes campos:

- Tarea_numTarea
- Tarea_Iteracion_numIteracion
- Tarea_Iteracion_Proyecto_idProyecto

```
/**
* Un tarea esta asociada a varios usuarios
* Debe ser un relacion bidireccional.
*/
@ManyToMany
@JoinTable(name="TareaUsuario",
    joinColumns={
        @JoinColumn(name="Tarea_numTarea",referencedColumnName="numTarea")
            ,
        @JoinColumn(name="Tarea_Iteracion_numIteracion",
            referencedColumnName="Iteracion_numIteracion"),
        @JoinColumn(name="Tarea_Iteracion_Proyecto_idProyecto",
            referencedColumnName="Iteracion_Proyecto_idProyecto")
    },
    inverseJoinColumns=@JoinColumn(name="Usuario_CURP",
        referencedColumnName="CURP")
)
public Set<Usuario> getRecursos() {
    return recursos;
}
```

Listing 6.20: Mapeado Tarea-Usuario

Figura 6.4: Tabla Tarea

- Usuario_CURP

Por lo tanto el mapeo se puede interpretar así. La tabla que permite la relación se llama *TareasUsuario*. Dicha tabla tiene una llave compuesta de tres atributos, *numTarea, Iteracion_numIteracion y Iteracion_Proyecto_idProyecto*. La columna que tiene el valor del campo *Tarea_numTarea* es la columna *numTarea* en la tabla Tarea. El valor de *Tarea_Iteracion_Proyecto_idProyecto*, referencia a la columna *Iteracion_Proyecto_idProyecto* en la tabla Tarea, *Tarea_Iteracion_Proyecto_idProyecto* referencia a la columna *Iteracion_Proyecto_idProyecto*. Al utilizar la anotación *joinColumns* especificamos que estos tres atributos conforman una llave foránea. Por otra parte, la columna *Usuario_CURP* es la llave foránea de la entidad objetivo de la relación y referencia a la columna *CURP* en la tabla Usuario.

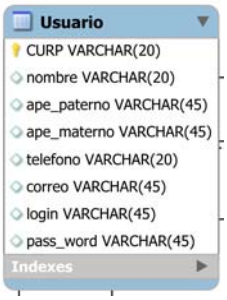

Figura 6.5: Tabla Usuario

6.2.6. Tarea.java

Ahora explicaremos el mapeado de la entidad Tarea. El primer aspecto a considerar es que tiene una llave primaria compuesta. Para realizar este tipo de mapeado creamos una clase que representa la llave primaria, la cual mostramos en 6.21.

```
package com.gestion.modelo.proyecto;
.... imports ...
@Embeddable
public class IdTarea implements Serializable{
  private long numTarea;
  private long numIteracion;
  private long idProyecto;
  private String idCadena;

  ... constructores

  public long getNumTarea() {
    return numTarea;
  }
  public void setNumTarea(long numTarea) {
    this.numTarea = numTarea;
  }
  @Column(name=``Iteracion_numIteracion")
  public long getNumIteracion() {
    return numIteracion;
  }
  public void setNumIteracion(long numIteracion) {
    this.numIteracion = numIteracion;
  }
  @Column(name=``Iteracion_Proyecto_idProyecto")
  public long getIdProyecto() {
    return idProyecto;
  }
  public void setIdProyecto(long idProyecto) {
    this.idProyecto = idProyecto;
  }
  @Transient
  public String getIdCadena() {
    return idProyecto+`` "+numIteracion+`` "+numTarea;
  }
  public void setIdCadena(String id) {
    this.idCadena = id;
  }
}
```

Listing 6.21: IdTarea.java

En este mapeado fue necesario especificar directamente el nombre de las columnas a las cuales van mapeados los atributos, también hay un atributo no persistente. Ahora

usaremos esta clase como identificador de la clase *Tarea.java*, tal como se ve en 6.22 en la cual no mostramos algunos atributos por claridad.

```
1  package com.gestion.modelo.proyecto;

3  .... imports

5  @Entity
   public class Tarea implements Serializable {
7    private IdTarea idTarea;
     private String nombreTarea;
9    private Date fechaInicio;
     private Date fechaFin;
11   private int hito;
     private int porcentaje;
13   private int pGroup;
     //private Tarea tareaPadre;
15   /**
      * Cardinalidad 0 o 1
17    */
     private List<Tarea> subtareas = new ArrayList<Tarea>();
19   private List<Tarea> tareaPadre = new ArrayList<Tarea>();
     private int pOpen;
21   private String pCaption;
     private int indice;
23   private Iteracion iteracion;
     private List<Tarea> predecesores = new ArrayList<Tarea>();
25   private Actividad actividad;
     private Set<Usuario> recursos = new HashSet<Usuario>();
27
     public Tarea() {
29     // TODO Auto-generated constructor stub
     }
31   @EmbeddedId
     public IdTarea getIdTarea() {
33     return idTarea;
     }
35 }
```

Listing 6.22: Tarea.java

Habrá observado que para indicar una llave compuesta se utiliza la anotación *@EmbeddedId* sobre el atributo *idTarea* que es de la clase *IdTarea*. El mapeado de la relación Tarea-Usuario ya la explicamos anteriormente. En cuando a la relación subtareas entre Tarea-Tarea la mostramos en 6.23.

La tabla que usamos para la relación es *PadreTareaHijo* y tiene lo siguientes campos:

- numTarea

```
@OneToMany
@JoinTable(name=``PadreTareaHijo",
    joinColumns={
    @JoinColumn(name=``numTarea",referencedColumnName=``numTarea"),
    @JoinColumn(name=``numProyecto",referencedColumnName=``
        Iteracion_Proyecto_idProyecto"),
    @JoinColumn(name=``numIteracion",referencedColumnName=``
        Iteracion_numIteracion")
    },
    inverseJoinColumns={
    @JoinColumn(name=``numTareaHijo",referencedColumnName=``numTarea"),
    @JoinColumn(name=``numProyectoH",referencedColumnName=``
        Iteracion_Proyecto_idProyecto"),
    @JoinColumn(name=``numIteracionH",referencedColumnName=``
        Iteracion_numIteracion")
}
)
public List<Tarea> getSubtareas() {
    return subtareas;
}
public void setSubtareas(List<Tarea> subtareas) {
    this.subtareas = subtareas;
}
```

Listing 6.23: Subtareas de tarea

- numIteracion
- numProyecto
- numTareaHijo
- numIteracionHijo
- numProyectoHijo

En el mapeado indicamos que la tabla de la relación es *PadreTareaHijo* y que sus columnas *numTarea, numProyecto, numIteracion* referencian a las columnas *numTarea, Iteracion_Proyecto_idProyecto* de la tabla *Tarea* respectivamente. Por otra parte, las columnas *numTareaHijo, numProyectoH* y *numIteracionH* referencian a las columnas *numTarea, Iteracion_Proyecto_idProyecto, Iteracion_numIteracion* respectivamente de la tabla Tarea.

Por otra parte, existe una relación tarea padre entre Tarea-Tarea, para eso realizamos el mapeado en 6.24.

El mapeado es muy similar a del subtareas, sólo que en esta ocasión es inverso. La propietaria de la relación es la tarea que tiene el papel de subtarea.

```
@OneToMany(fetch=FetchType.EAGER)
@JoinTable(name=``PadreTareaHijo",
    joinColumns={
    @JoinColumn(name=``numTareaHijo",referencedColumnName=``numTarea"),
    @JoinColumn(name=``numProyectoH",referencedColumnName=``
        Iteracion_Proyecto_idProyecto"),
    @JoinColumn(name=``numIteracionH",referencedColumnName=``
        Iteracion_numIteracion")

    },
    inverseJoinColumns={
    @JoinColumn(name=``numTarea",referencedColumnName=``numTarea"),
    @JoinColumn(name=``numProyecto",referencedColumnName=``
        Iteracion_Proyecto_idProyecto"),
    @JoinColumn(name=``numIteracion",referencedColumnName=``
        Iteracion_numIteracion")
}
)
public List<Tarea> getTareaPadre() {
    return tareaPadre;
}
public void setTareaPadre(List<Tarea> padre) {
    this.tareaPadre = padre;
}
```

Listing 6.24: Relación Tarea(Padre)-Tarea(Hijo)

Ahora mapearemos la relación predecesores de una tarea 6.25.

El mapeado es muy similar a las anteriores. Con la diferencia de que es otra tabla la que se usa para la relación, la tabla *TareaPredecesor*.

También hay una relación entre Tarea-Actividad, la cual es uno a uno, y se muestra en 6.26.

La relación Tarea-Usuario ya fue explicada anteriormente.

```
@OneToMany
@JoinTable(
    name="TareaPredecesor",
    joinColumns={
        @JoinColumn(name="numTarea",referencedColumnName="numTarea"),
        @JoinColumn(name="numIteracion",referencedColumnName="
            Iteracion_numIteracion"),
        @JoinColumn(name="numProyecto",referencedColumnName="
            Iteracion_Proyecto_idProyecto")
    },
    inverseJoinColumns={
        @JoinColumn(name="numTareaPredecesor",referencedColumnName="
            numTarea"),
        @JoinColumn(name="numIteracionP",referencedColumnName="
            Iteracion_numIteracion"),
        @JoinColumn(name="numProyectoP",referencedColumnName="
            Iteracion_Proyecto_idProyecto")
    }
)
public List<Tarea> getPredecesores() {
    return predecesores;
}
public void setPredecesores(List<Tarea> predecesores) {
    this.predecesores = predecesores;
}
```

Listing 6.25: Tarea Predecesor de Tarea

```
@OneToOne
@JoinColumn(name="Actividad_idActividad")
public Actividad getActividad() {
    return actividad;
}
public void setActividad(Actividad actividad) {
    this.actividad = actividad;
}
```

Listing 6.26: Tarea-Actividad

6.2.7. Artefacto.java

El mapeado para esta entidad se muestra en 6.27, se ha omitido el mapeado de algunos atributos.

```
package com.gestion.modelo.proyecto;

... imports ...

@Entity
public class Artefacto implements Serializable{
  private String nombre;
  private String descripcion;
  private String urlDescripcion;
  private Rol rol;
  private String proposito;
  private String timing;
  private String tailoring;
  private String representacionUML;

  public Artefacto(){

  }
  @Id
  public String getNombre() {
    return nombre;
  }

  ...

  @OneToOne
  @JoinColumn(name="Rol_nombreRol")
  public Rol getRol() {
    return rol;
  }
  public void setRol(Rol rol) {
    this.rol = rol;
  }
}
```

Listing 6.27: Artefacto.java

6.2.8. EjemplarArtefacto.java

Esta entidad tiene una llave compuesta que hemos mapeado en la clase IdEjemplar, mostrado en 6.28.

```
1  package com.gestion.modelo.ejemplarArtefacto;
2
3  import java.io.Serializable;
   import javax.persistence.Column;
5
   public class IdEjemplarArtefacto implements Serializable {
7    private long idProyecto;
     private String nombreArtefacto;
9
     public IdEjemplarArtefacto(long idProyecto, String nombreArtefacto){
11     this.idProyecto = idProyecto;
       this.nombreArtefacto = nombreArtefacto;
13   }
14
15
     @Column(name="Proyecto_idProyecto")
17   public long getIdProyecto() {
       return idProyecto;
19   }
     public void setIdProyecto(long idProyecto) {
21     this.idProyecto = idProyecto;
     }
23   @Column(name="Artefacto_nombre")
     public String getNombreArtefacto() {
25     return nombreArtefacto;
     }
27   public void setNombreArtefacto(String nombreArtefacto) {
       this.nombreArtefacto = nombreArtefacto;
29   }
   }
```

Listing 6.28: IdEjemplar.java

Ahora el mapeo de la entidad EjemplarArtefacto en 6.29.

```
   package com.gestion.modelo.ejemplarArtefacto;
2
   ...  imports ....
4
   @Entity
6  public class EjemplarArtefacto implements Serializable{
     private IdEjemplarArtefacto idEjemplar;
8    private String nombreArchivo;
     private String tipoContenido;
10   private Artefacto artefacto;
```

```
  private Blob contenidoArchivo;
12  private Proyecto proyecto;
  private Date fechaCreacion;
14
  public EjemplarArtefacto(){
16
  }
18  @EmbeddedId
  public IdEjemplarArtefacto getIdEjemplar() {
20    return idEjemplar;
  }
22
  ...
24
  @OneToOne
26  @JoinColumn(name="Artefacto_nombre",insertable=false, updatable=false
      )
  public Artefacto getArtefacto() {
28    return artefacto;
  }
30  public void setArtefacto(Artefacto artefacto) {
    this.artefacto = artefacto;
32  }
  @Lob
34  public Blob getContenidoArchivo() {
    return contenidoArchivo;
36  }
  public void setContenidoArchivo(Blob contenidoArchivo) {
38    this.contenidoArchivo = contenidoArchivo;
  }
40  @OneToOne
  @JoinColumn(name="Proyecto_idProyecto",insertable=false, updatable=
      false)
42  public Proyecto getProyecto() {
    return proyecto;
44  }.
    ....
46 }
```

Listing 6.29: Ejemplar.java

Lo nuevo en esta ocasión es el mapeo de un atributo de tipo *java.sql.Blob*, el cual se hace con la anotación *@Lob*.

6.2.9. Plantilla.java

Por último mapeamos la entidad Plantilla, como se ve en 6.30.

```
package com.gestion.modelo.plantilla;
... imports ..
public class Plantilla implements Serializable {
  private IdPlantilla id;
  private String nombreArchivo;
  private String tipoContenido;
  private Blob contenido;
  private Date fechaCreacion;
  private Artefacto artefacto;
  private Proyecto proyecto;

  @EmbeddedId
  public IdPlantilla getId() {
    return id;
  }
  public void setId(IdPlantilla id) {
    this.id = id;
  }

  ...

  @OneToOne
  @JoinColumn(name="Artefacto_nombre",insertable=false, updatable=false
      )
  public Artefacto getArtefacto() {
    return artefacto;
  }
  public void setArtefacto(Artefacto artefacto) {
    this.artefacto = artefacto;
  }
  @OneToOne
  @JoinColumn(name="Proyecto_idProyecto", insertable=false, updatable=
      false)
  public Proyecto getProyecto() {
    return proyecto;
  }
  public void setProyecto(Proyecto proyecto) {
    this.proyecto = proyecto;
  }
}
```

Listing 6.30: Plantilla.java

Esta entidad tiene una llave compuesta del tipo *IdPlantilla*, la cual se mapeó como se observa en 6.31.

```
package com.gestion.modelo.plantilla;
... imports ..
public class IdPlantilla implements Serializable{
  private long numProyecto;
  private String nombreArtefacto;

  @Column(name=``Proyecto_idProyecto")
  public long getNumProyecto() {
    return numProyecto;
  }
  public void setNumProyecto(long numProyecto) {
    this.numProyecto = numProyecto;
  }
  @Column(name=``Artefacto_nombre")
  public String getNombreArtefacto() {
    return nombreArtefacto;
  }
  public void setNombreArtefacto(String nombreArtefacto) {
    this.nombreArtefacto = nombreArtefacto;
  }
}
```

Listing 6.31: IdPlantilla.java

Hasta ahora, hemos explicado el proceso de implementación para las clases persistentes. Sin embargo, aún nos falta implementar las pantallas y las clases que menejarán los eventos asociados a las mismas. Esto lo abordaremos en los capítulos siguientes.

Capítulo 7

Implementación de Pantallas del sistema

7.1. Menú Principal

En el capítulo 4 en la sección 4.10 relacionado con el mapa de navegación presentamos el menú principal que se muestra en la figura 7.1.

Figura 7.1: Menú Principal

De aquí en adelante desglosaremos las implementación de cada opción del menú.

7.2. Menú Inicio

Este menú tendrá dos pantallas asociadas, una de bienvenida al sistema y otra que le permitirá al usuario validarse en el sistema. En la figura 7.2 mostramos la pantalla de validación que se implementó.

Figura 7.2: Pantalla de Validación del usuario

7.3. Menú Proyecto

De acuerdo a los requerimientos especificados en este menú el usuario tendrá las pantallas que muestra la figura 7.3.

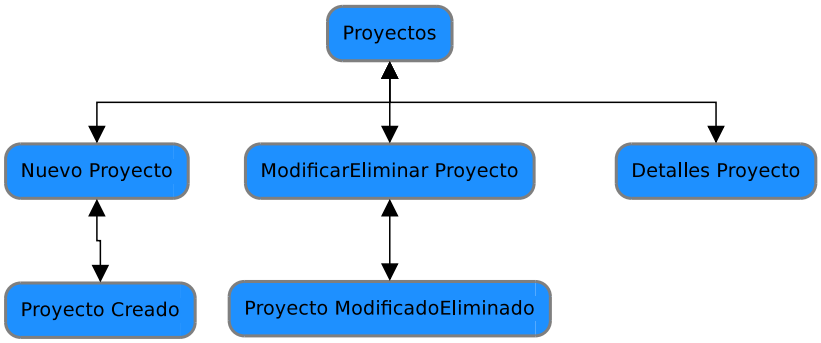

Figura 7.3: Pantallas del menú Proyecto

Como se puede apreciar, la primera pantalla a mostrarse es "Proyectos", en la cual se muestran los proyectos en los que el usuario está participando. En esta pantalla el usuario debe elegir un proyecto sobre el cual trabajar. La pantalla implementada se muestra en 7.4.

Figura 7.4: Pantalla de proyectos del usuario

Desde la pantalla 7.4 el usuario puede ir a una pantalla que muestra información detallada del proyecto, esta pantalla implementada se muestra en 7.5.

Figura 7.5: Pantalla que muestra información detallada del proyecto

Por otra parte desde la pantalla 7.4 el usuario puede solicitar modificar los datos de un proyecto, para dicho propósito se implementó la pantalla 7.6.

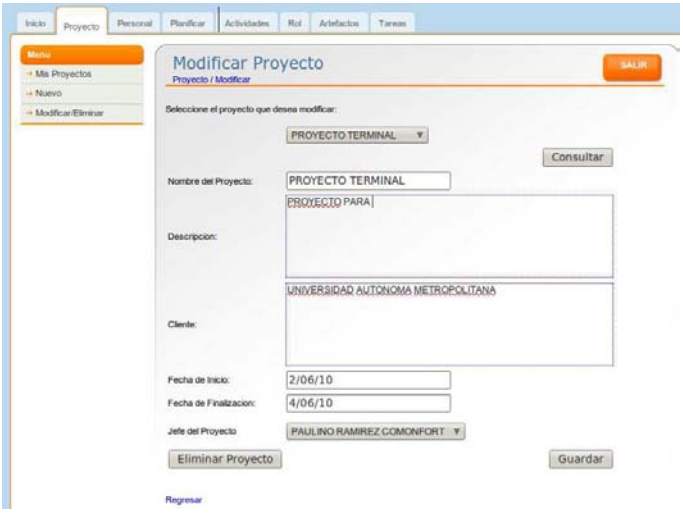

Figura 7.6: Pantalla para modificar los datos de un proyecto

Por último en este menú el usuario tiene la opción de crear un nuevo proyecto pulsando sobre *Nuevo*, tras lo cual se desplegará la pantalla mostrada en 7.7.

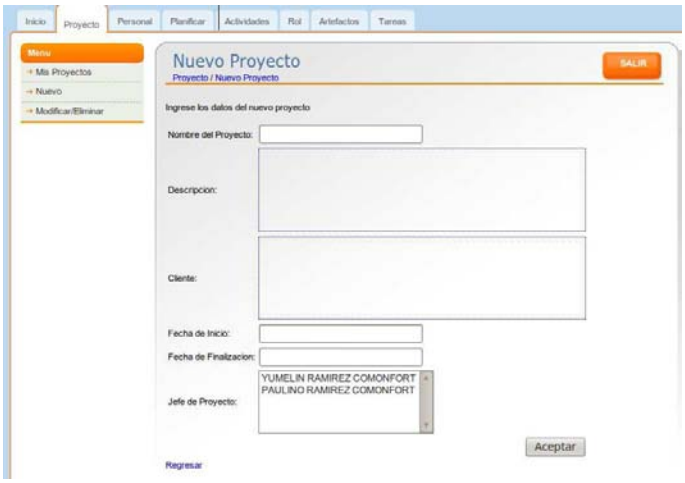

Figura 7.7: Pantalla para registrar un nuevo proyecto

7.4. Menú Personal

Para este menú se crearon las pantallas que muestra 7.8.

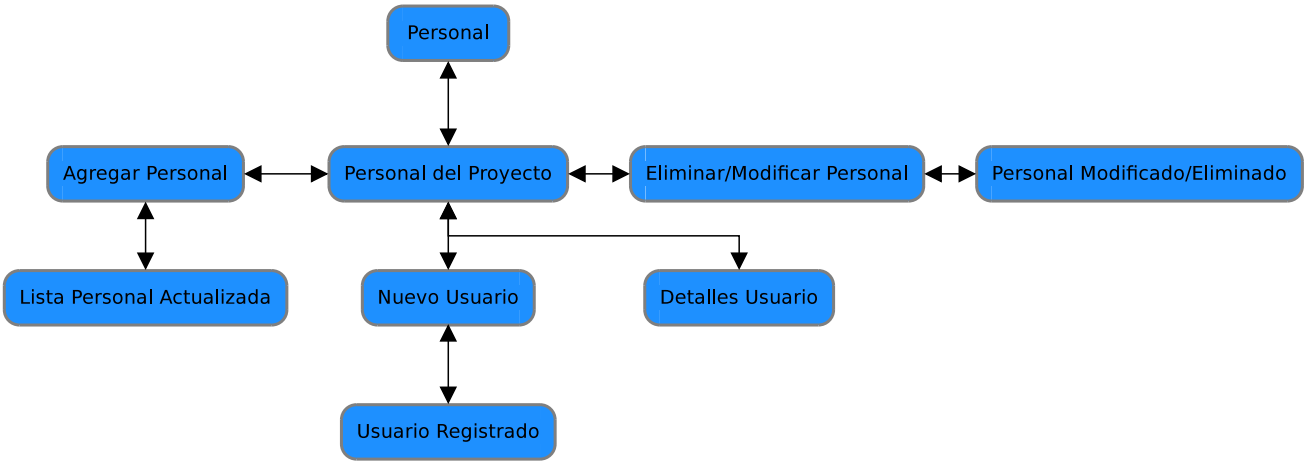

Figura 7.8: Pantallas del menú Personal

La pantalla que se presenta por defecto en este menú es la pantalla de *Personal del Proyecto*. Esta pantalla se muestra en 7.9. Desde esta pantalla al dar clic en *Detalles* se despliega una pantalla que muestra la información detallada del usuario.

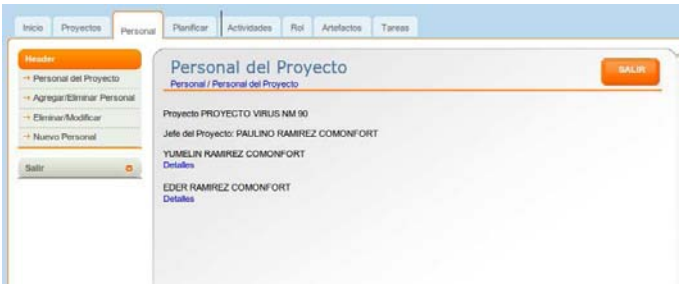

Figura 7.9: Pantalla que muestra el personal del proyecto

Con la opción *Agregar/Eliminar Personal* se muestra la pantalla de la figura 7.10. Donde por un lado está una lista del personal no asignado y por el otro al personal que está asignado actualmente al proyecto.

Por otra parte se pueden modificar los datos de un usuario. En la pantalla *Modificar/Eliminar Personal* mostrada enla figura 7.11.

Figura 7.10: Pantalla para agregar o quitar personal al proyecto

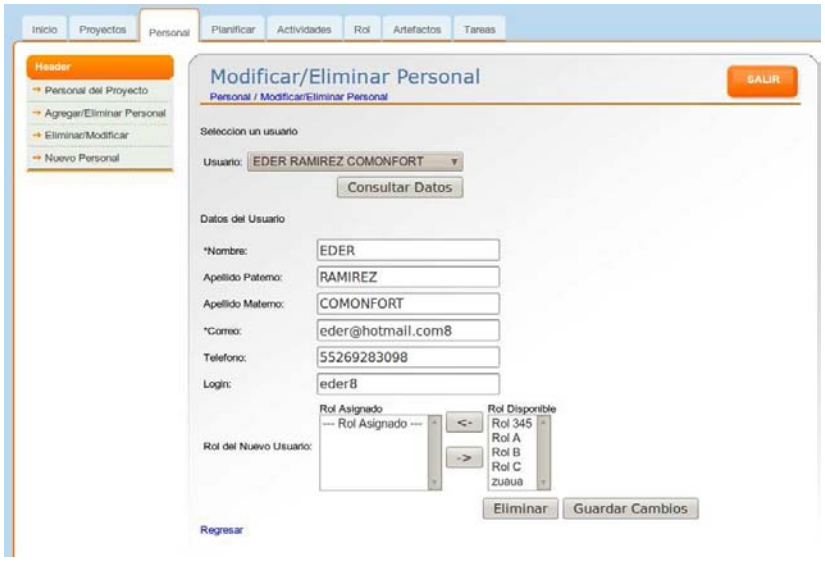

Figura 7.11: Pantalla para modificación o eliminación de un usuario

7.5. Menú Planificar

Las pantallas a implementar se presentan gráficamente en 7.12. Al dar clic en este menú se presentará una pantalla que muestra las iteraciones del proyecto que se haya elegido. Esta pantalla la observamos en la figura 7.13.

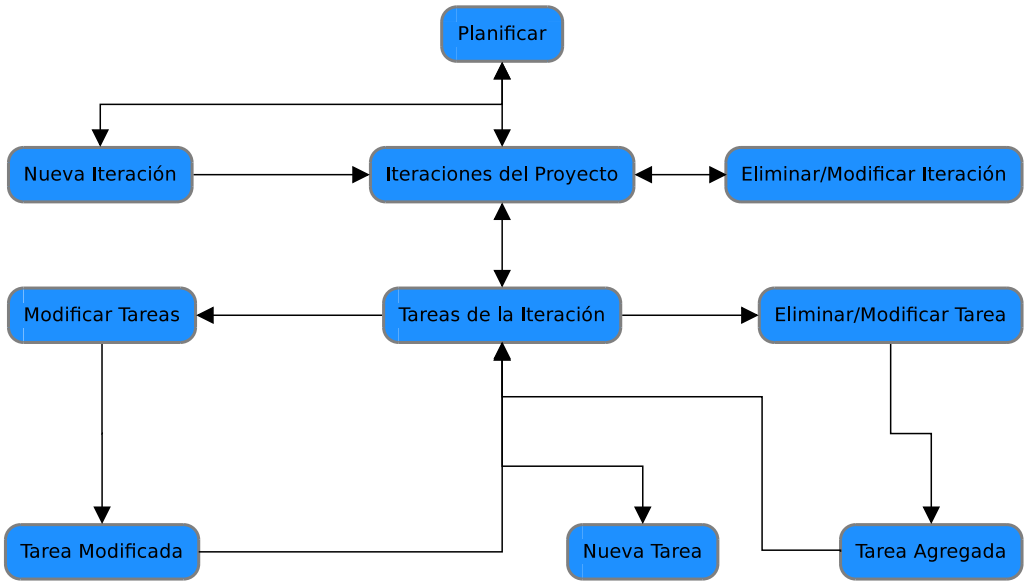

Figura 7.12: Navegación en Planificar

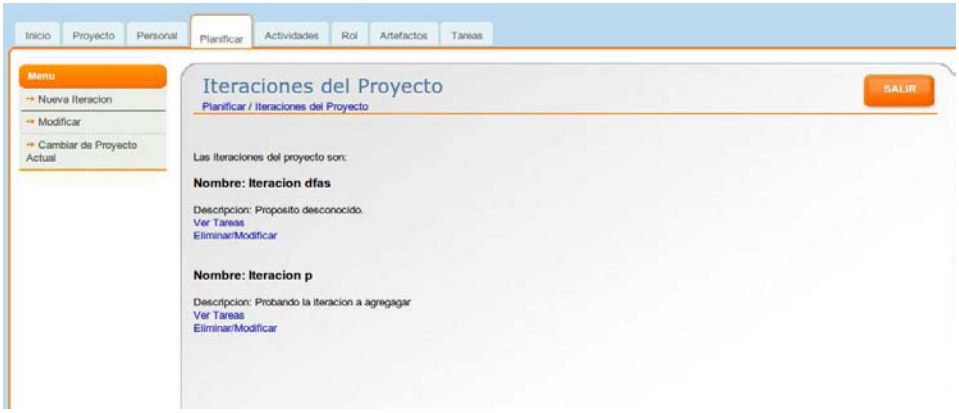

Figura 7.13: Pantalla para visualizar las iteraciones del proyecto.

En la pantalla 7.13 se observa que para cada iteración se pueden consultar las tareas haciendo clic sobre *Ver Tareas*, tras lo cual se presentará una pantalla que muestra las tareas programadas agrupadas en las 7 disciplinas que define RUP, tal como se ve en 7.14. Además en esta pantalla se dibuja el diagrama de Gannt de acuerdo a la programación de las tareas. Por otra parte, se puede modificar la información de

una iteración haciendo clic en *Modificar/Eliminar Iteración*, con lo que se presenta la pantalla de la figura 7.15 que permite cambiar la información asociada a una iteración.

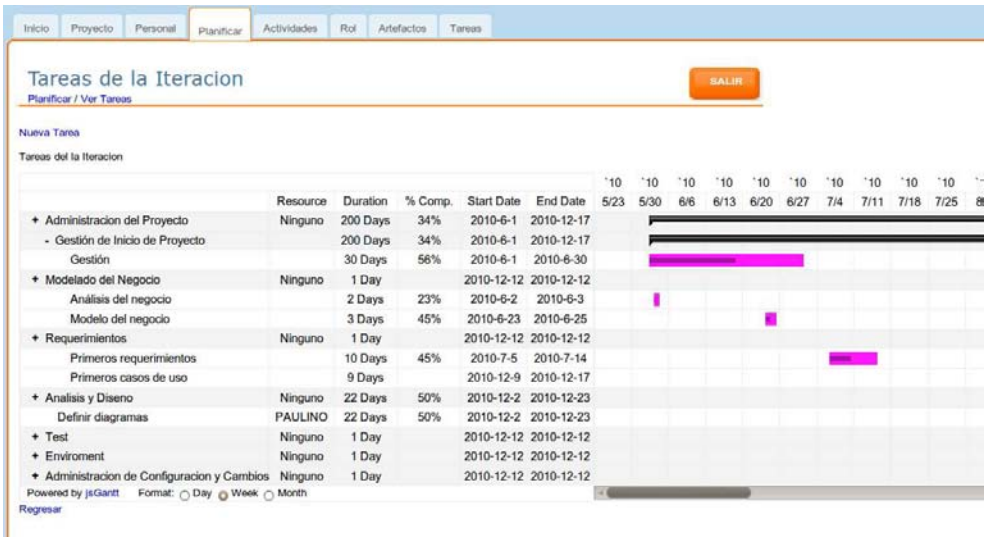

Figura 7.14: Pantalla que muestra las tareas de la iteracón.

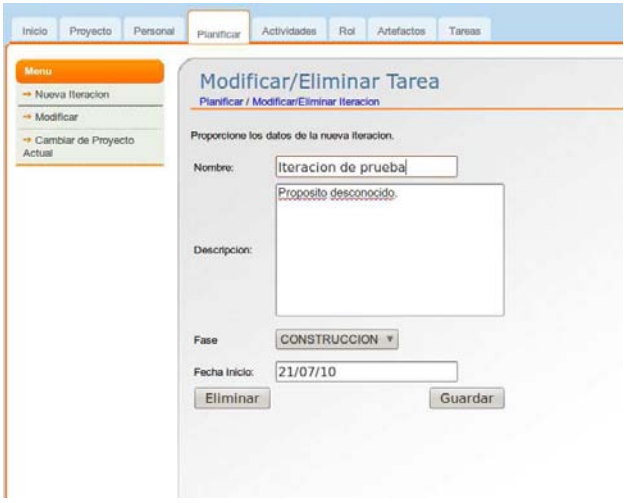

Figura 7.15: Pantalla para la modificación o eliminación de una iteración.

Para agregar una tarea a la iteración basta con hacer clic sobre *nuevaTarea* presentandose una pantalla donde se tiene que capturar la información de la nueva tarea, tales como el nombre, fecha de inicio, fecha de finalización, disciplina, si es subtarea de alguna otra tarea, las tareas que le preceden y el personal al cual será asignado, todo en la pantalla de la figura 7.16.

Figura 7.16: Pantalla para agregar tarea a la iteración.

7.6. Menú Actividades

En este meú son necesarias las pantallas representadas en 7.17.

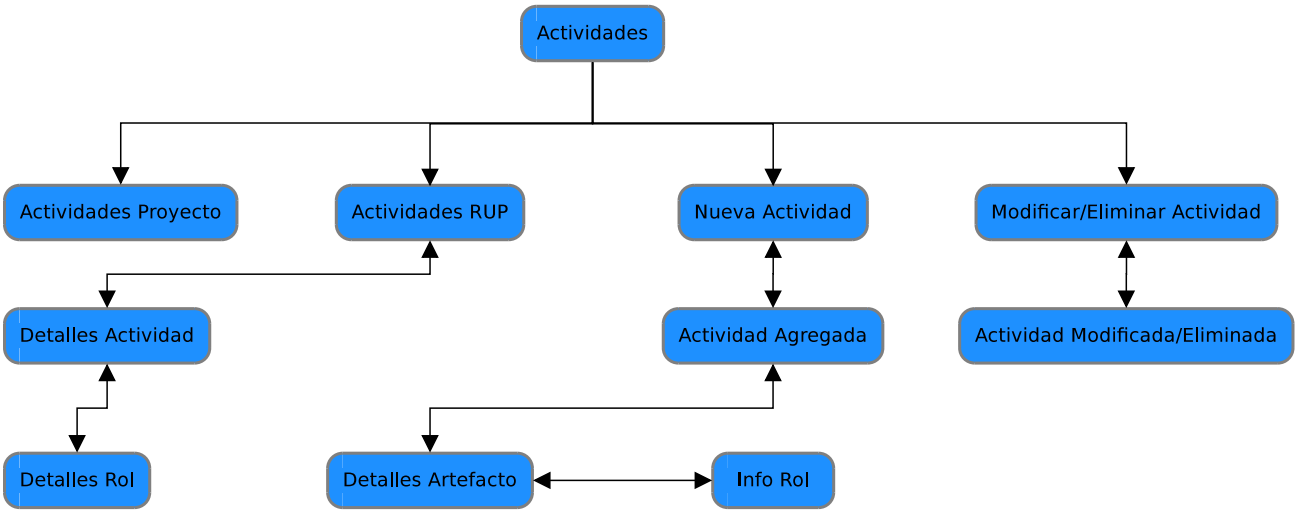

Figura 7.17: Navegación en Actividades.

En este menú la pantalla que se presentará por defecto mostrará todas las actividades definidas por RUP, justo debajo de cada actividad, se mostrará el rol del usuario que puede realizar dicha actividad. Sobre el nombre de cada actividad se puede hacer clic para ver información detallada de la actividad, lo mismo en el caso del rol. También desde esta pantalla se puede acceder a una pantalla que permitirá la modificación de una actividad. Esta pantalla se muestra en la figura 7.18.

Figura 7.18: Pantalla para listar las actividades definidas por RUP.

Por otra parte, la pantalla que muestra información detallada de una actividad se muestra en la figura 7.19.

Figura 7.19: Información detallada de una actividad.

En la figura 7.20 podemos observar la pantalla que se presenta cuando el usuario haga clic en *Modificar/Eliminar*, en la cual puede agregar o completar la información relacionada con el propósito de una actividad, los artefactos de entrada y salida, el rol, la disciplina a la cual pertenece la actividad, u otra información general sobre la actividad.

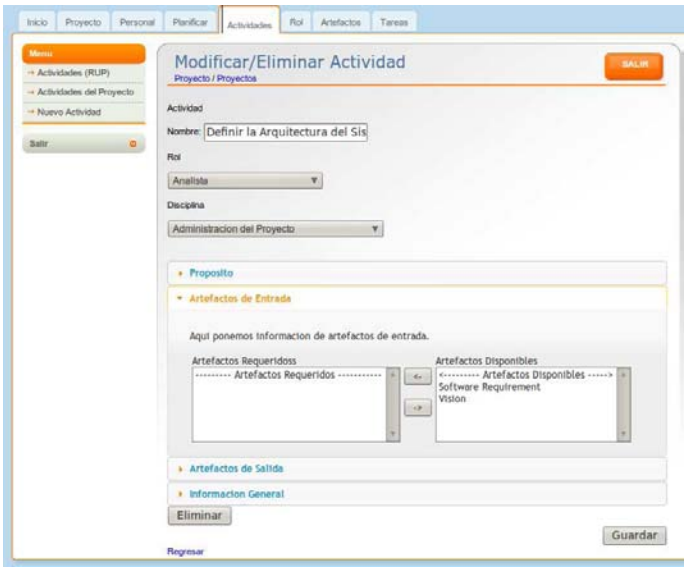

Figura 7.20: Pantalla de modificación o eliminación de una actividad.

En ocasiones será necesario agregar una actividad al sistema, para eso se ha creado la pantalla de la figura 7.21, en la cual el usuario debe proporcionar todos los datos necesarios para poder registrar la actividad en el sistema.

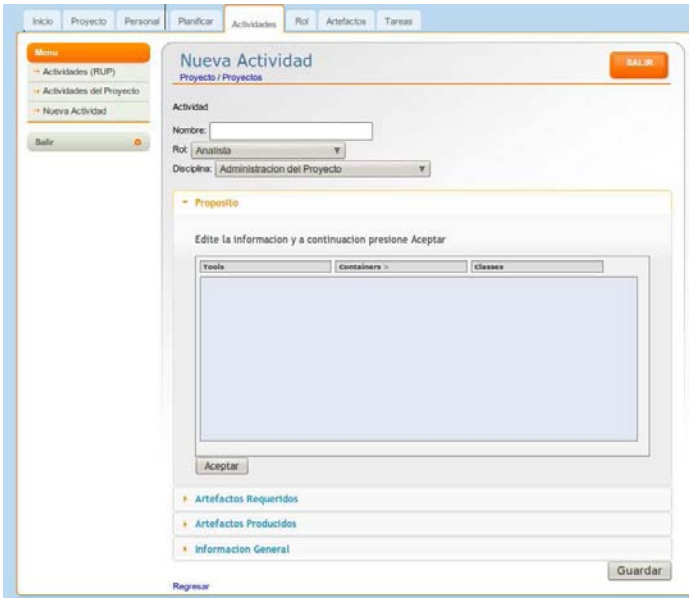

Figura 7.21: Pantalla para agregar una actividad.

7.7. Menú Rol

Para este menú hemos definido cuatro pantallas. Una que muestra todos los roles definidos en el sistema, otra para la modificación o eliminación de un rol, otra para ver los detalles del rol y una última para agregar un nuevo rol. La primera pantalla se muestra en la figura 7.22.

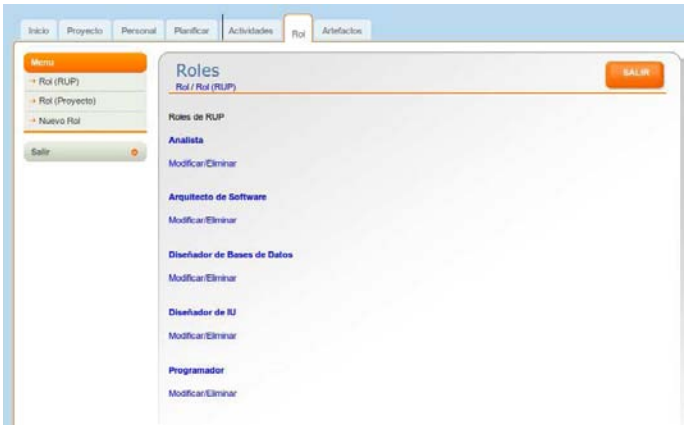

Figura 7.22: Pantalla que muestra los roles definidos por RUP.

La pantalla de modificación o eliminación de un rol se aprecia en 7.23.

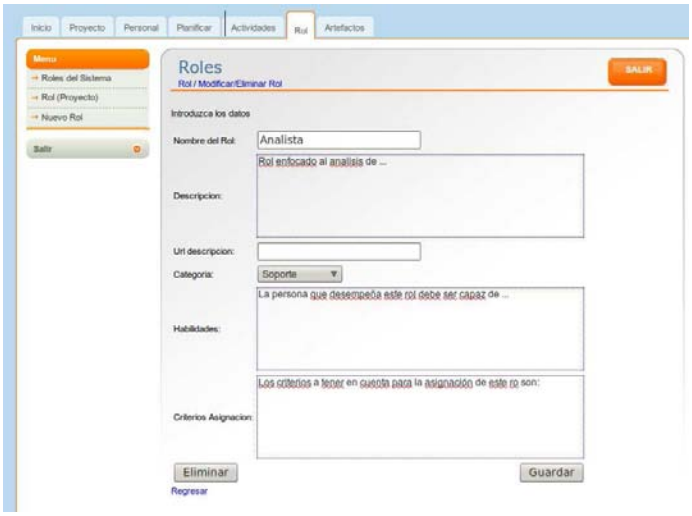

Figura 7.23: Pantalla para modificación o eliminación de un rol.

Por último la pantalla para agregar un nuevo rol se muestra en la figura 7.24 en la cual el usuario tiene que proporcionar los datos y después hacer clic en *Registrar*.

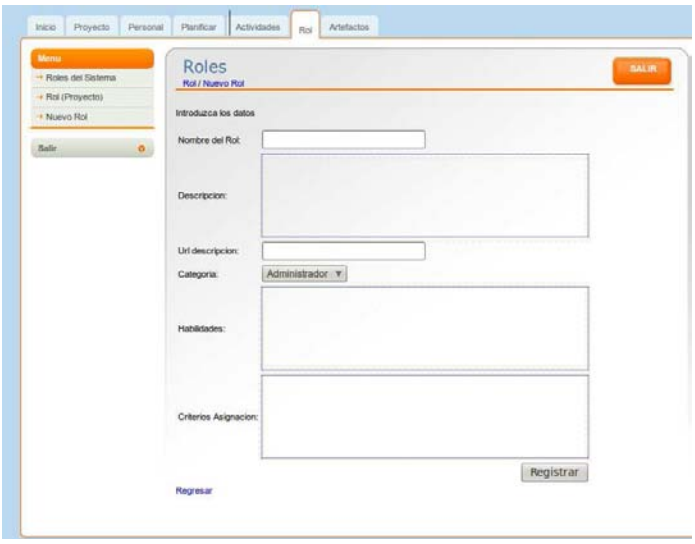

Figura 7.24: Pantalla para agregar un nuevo rol.

7.8. Menú Plantillas

En el menú Plantillas se implementaron las pantallas 7.25.

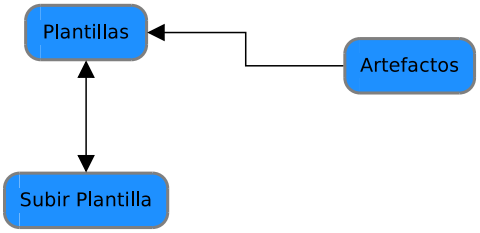

Figura 7.25: Navegación en Plantillas

Estas pantallas son complemento de la administración de artefactos, por lo que se mostrarán cuando presentemos las pantallas del menú Artefactos.

7.9. Menú Tareas

Para el manejo de tareas de cada usuario se han creado las pantallas que muestra la figura 7.26.

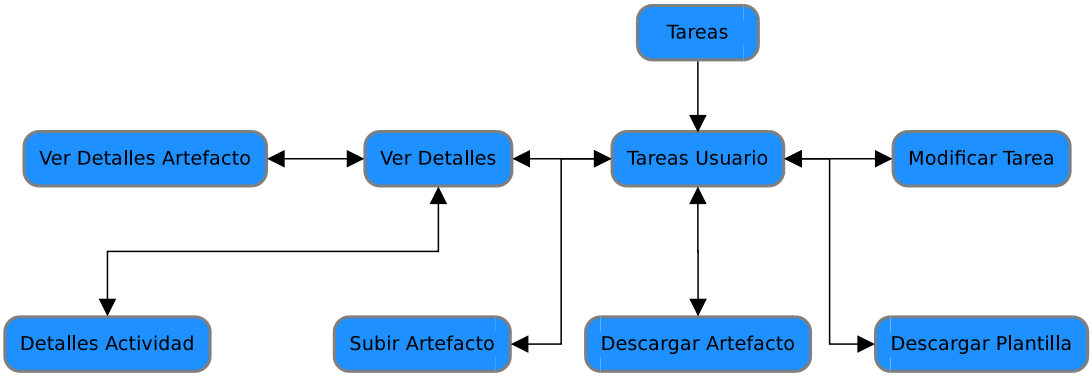

Figura 7.26: Navegación en Tareas

La pantalla a presentarse por defecto muestra el listado de tareas que se le han asignado al usuario. La implementación de dicha pantalla la podemos ver en la figura 7.27. En esta pantalla se muestra información resumida de cada tarea, sin embargo, hay dos enlaces que permiten obtener información detallada. Primeramente se tiene un enlace en la palabra remarcada de azul *Artefactos*, la cual mostrará otra pantalla con información detallada de los artefactos requeridos para realizar la tarea y de los que se deben producir al finalizar la misma. Esta última pantalla se puede observar en la figura 7.28.

Figura 7.27: Lista de tareas del usuario.

Figura 7.28: Artefactos requeridos y producidos por la tarea.

En la pantalla 7.28, en el caso de los artefactos que se van a producir el usuario tien la opción *Subir Artefacto*, con lo cual se despliega una pantalla que permite al usuario subir un archivo, que en este caso sería el artefacto realizado. Lo anterior se puede visualizar en la figura 7.29. Si el archivo es subido correctamente se mostrará un mensaje indicando el éxito, en caso contrario un mensaje de error.

Por otra parte, se tiene la opción de descargar el archivo de un artefacto ya realizado, esto en el enlace *Descargar Artefacto*, el cual presenta la pantalla de la figura 7.30.

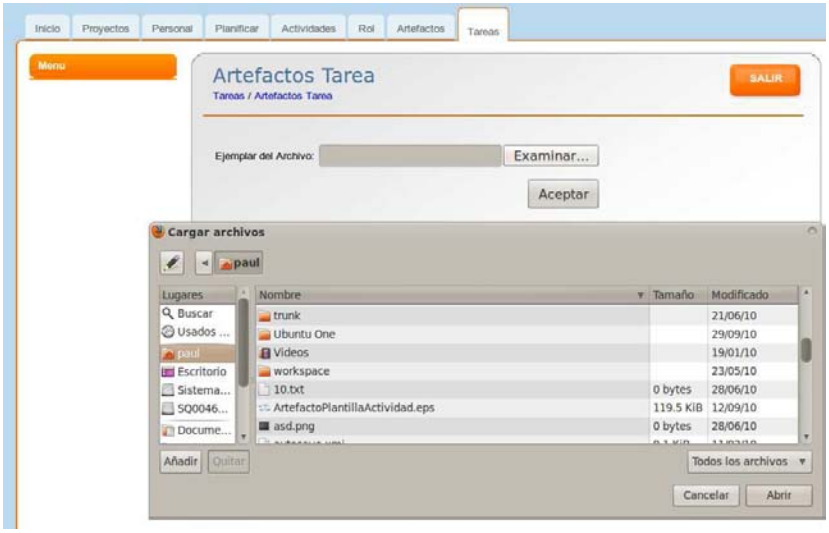

Figura 7.29: Pantalla para subir un archivo al sistema.

Figura 7.30: Descargando un archivo correspondiente a un artefacto.

7.10. Menú Artefactos

En este menú se han definido tres pantallas: Nuevo Artefacto, Artefactos (RUP) y Modificar/Eliminar Artefacto. La primera pantalla es para que el usuario dé de alta un artefacto en el sistema. El segundo realiza un listado de los artefactos definidos en el sistema y el último permite modificar o eliminar un artefacto.

La pantalla para agregar un artefacto se muestra en la figura 7.31.

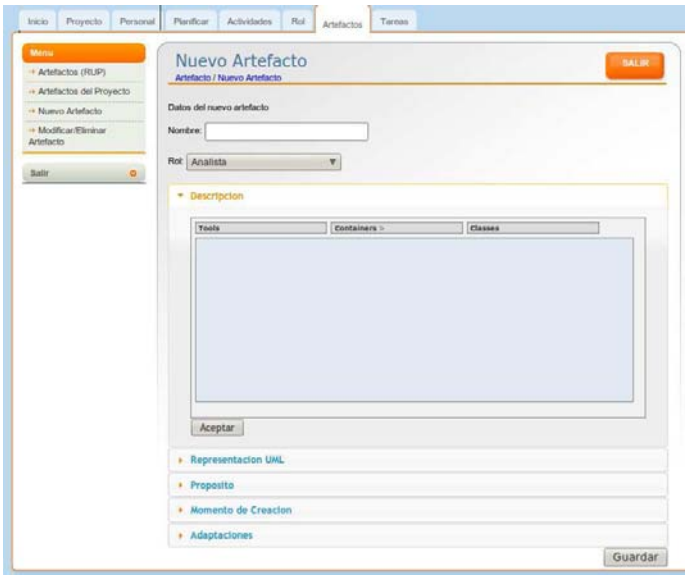

Figura 7.31: Pantalla para agregar un artefacto.

En la figura 7.32 es para modificar o eliminar un artefacto.

Por otra parte para observar los detalles de una artefacto se creó la pantalla 7.33.

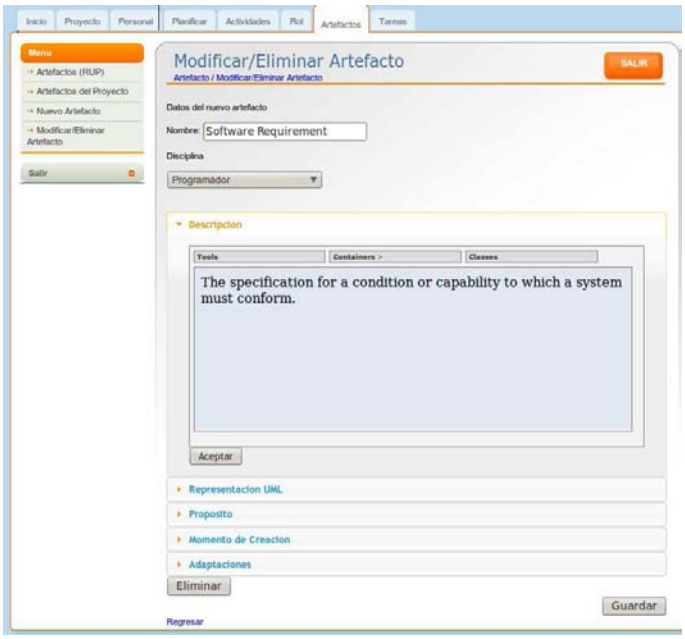

Figura 7.32: Pantalla para modificación o eliminación de artefacto.

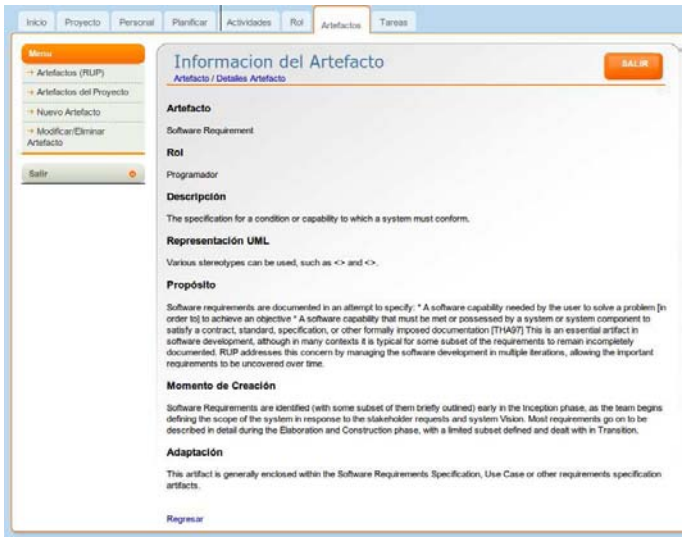

Figura 7.33: Pantalla Detalles de artefacto.

7.11. Definición de Tiles

Todas las pantallas anteriores ha sido definidas utilizando Tiles. Los Tiles son una especie de plantillas que se utilizan para definir el Layout, el cual puede reutilizarse. La utilización de Tiles nos permite definir en el Layout qué elementos serán estáticos y cuáles serán dinámicos. La integración de tiles con Struts ya la hemos explicado en la sección de tecnologías utilizadas. Ahora nos centraremos en la definición de los tiles.

Primeramente tenemos que definir la plantilla, esto lo hemos hecho en un archivo que nombramos *estructuraGeneral.jsp*, cuyo contenido se muestra en el código 7.1.

```
1 <%@ page language=``java" contentType=``text/html;
        charset=UTF-8"    pageEncoding=``UTF-8"%>
3 <%@ taglib prefix=``s" uri=``/struts-tags" %>
  <%@ taglib uri=``http://tiles.apache.org/tags-tiles" prefix="tiles"%>
5
  <!DOCTYPE html PUBLIC ``-//W3C//DTD HTML 4.01 Transitional//EN"
7       ``http://www.w3.org/TR/html4/loose.dtd">

9 <html>
  <head>
11 <meta http-equiv=``Content-Type" content=``text/html; charset=UTF-8">
  <title><tiles:getAsString name=``pageTitle"/></title>
13
  </head>
15 <body>

17 <div id=``main">
     <!-- header -->
19   <tiles:insertAttribute name=``header"/>

21   <!-- middle -->
     <tiles:insertAttribute name=``middle"/>
23
     <!-- footer -->
25   <tiles:insertAttribute name=``footer"/>
  </div>
27
  </body>
29 </html>
```

Listing 7.1: Plantilla utilizada en la aplicación

Como se puede apreciar las partes dinámicas de la vista son *pageTitle*, *header*, *middle* y *footer*. Con esto mantenemos la misma estructura de nuestras vistas y variamos sólo aquellas partes que se requieran. Ahora tenemos que definir las vistas asignado valores a los atributos dinámicos *pageTitle*, *header*, *middle* y *footer*. En las

tablas se muestran todas las vistas que hemos definido y una breve descripción de su propósito. Ver tablas 7.1 y 7.2.

Tile	**Descripción**
Inicio	Pantalla con mensaje de bienvenida al usuario.
Login	Pantalla desde donde el usuario inicia sesión.
Error	Pantalla con mensaje de error.
PersonalProyecto	Pantalla que muestra la lista de usuarios que participan en el proyecto actual.
DetallesUsuario	Pantalla que muestra información detallada de un usuario.
AsignarParticipantesProyecto	Permite agregar o quitar usuarios al proyecto actual.
NuevoUsuario	Pantalla con los campos para capturar y registrar los datos de un nuevo usuario.
ListaPersonalActualizada	Pantalla a mostrarse después de quitar o agregar un usuario al proyecto.
UsuarioRegistrado	Muestra los datos del usuario recién agregado.
ModificarEliminarUsuario	Permite modificar los datos de un usuario o eliminarlo.
UsuarioActualizado	Muestra los datos actualizados del usuario modificado.
UsuarioEliminado	Muestra mensaje de confirmación del usuario eliminado.
RegistrarPersonal	Pantalla para agregar un nuevo usuario al proyecto actual, con posibilidad para agregarlo a otro proyecto.
ProyectosUsuario	Muestra un listado de los proyectos en los que el usuario actual está participando.
ModificarEliminarProyecto	Permite modificar información asociada a un proyecto.
ProyectoModificadoEliminado	Pantalla que muestra mensaje de confirmación al eliminar un proyecto.
NuevoProyecto	Muestra los campos para capturar los datos del nuevo proyecto.
ProyectoRegistrado	Muestra la confirmación al registrar un proyecto nuevo.
DetallesProyecto	Muestra información detallada de un proyecto.
RolesSistema	Muestra los roles definidos en el sistema.
InfoRol	Detalles de un rol.
NuevoRol	Pantalla para capturar y registrar un nuevo rol.
ModificarEliminarRol	Permite modificar la información asociada a un rol o eliminar el rol.
RolAgregado	Mensaje de confirmación al agregar un nuevo rol.
Permisos	Permite visualizar permisos para un rol.
NuevaAccion	Agregar una acción en el sistema.
ModificarEliminarAccion	Permite modificar o eliminar una acción en el sistema.

Tabla 7.1: Lista de Tiles

Tile	Descripción
Iteraciones	Muestra la lista de iteraciones del proyecto actual.
Tareas	Muestra las tareas de una iteración en un diagrama de Gantt.
DetallesTarea	Visualización de los detalles de la tarea seleccionada.
NuevaIteracion	Muestra los campos para capturar los datos de una iteración y registrarla en el sistema.
ModificarEliminarIteracion	Permite la modificación o eliminación de una iteración.
NuevaTarea	Muestra los campos para registrar una nueva tarea.
ModificarTarea	Permite modificar la información asociada a una tarea.
TareaModificada	Mensaje de confirmación de la tarea modificada.
TareaRegistrada	Mensaje de confirmación después de registrar una tarea.
TareaEliminada	Mensaje de confirmación al eliminar una tarea.
ArtefactosSistema	Listado de los artefactos registrados en el sistema.
NuevoArtefacto	Pantalla para registrar un nuevo artefacto en el sistema.
ArtefactoAgregado	Mensaje de confirmación al agregar un nuevo artefacto.
DetallesArtefacto	Información detallada del artefacto.
DetallesRol	Información detallada del rol.
ModificarEliminarArtefacto	Permite realizar cambios de la información del artefacto.
ArtefactoModificado	Mensaje de confirmación después de modificar un artefacto.
ArtefactoEliminado	Mensaje de confirmación después de elimiinar un artefacto.
ActividadesRup	Muestra un listado de las actividades registradas en el sistema.
DetallesActividad	Información detallada de la actividad seleccionada.
ModificarEliminarActividad	Pantalla para modificación o eliminación de una actividad.
ActividadModificada	Mensaje de confirmación después de modificar una actividad.
NuevaActividad	Pantalla para registrar una nueva actividad en el sistema.
ActividadAgregada	Mensaje de confirmación después de agregar una actividad.
TareasUsuario	Listado de las tareas del usuario actual.
DetallesTareaUsuario	Información detallada de la tarea elegida.
VerDetallesActividad	Información de la actividad elegida.
VerDetallesRol	Información del rol elegido.
ArtefactosTarea	Listado de artefactos asociados a la tarea elegida.
ElegirArchivo	Pantalla para subir un archivo del artefacto elegido.
SubidoExitosamente	Mensaje de confirmación después de subir un archivo.

Tabla 7.2: Lista de Tiles

Capítulo 8

Implementación del Modelo y Controlador de la aplicación

8.1. El Modelo en el patro MVC

Como se comentó en la sección de tecnologías utilizadas, Struts, implementa el patrón Modelo-Vista-Controlador. En el capítulo anterior cubrimos todo lo referente a la vista, es decir, la interfaz de usuario. El controlador, que es el encargado de recibir los eventos de la interfaz y comunicarlo al modelo correspondiente se implementa en Struts mediante el filtro FilterDispatcher. Por lo anterior a continuación describiremos la implementación del controlador y después la implementación del modelo.

8.2. Controlador

Para poder utilizar el controlador FilterDispatcher se debe de editar el archivo *web.xml*. Este archivo contiene la configuración de la aplicación web. Primeramente se tiene que agregar el filtro que gestiona las peticiones a Struts 2, tal como se observa en 8.1. Con lo anterior estamos indicando a Apache Tomcat que toda petición debe ser interceptada y gestionada por *FilterDispatcher*, el cual es el controlador de la aplicación web.

Posteriormente tenemos que editar el archivo *struts.xml*, el cual tiene la configuración de Struts. Es en este archivo indicamos todas las acciones (o peticiones) a las cuales responderá nuestro sistema, el cual es finalmente el mapeado de la aplicación. Con dicho mapeado la aplicación sabrá qué modelo manejará la acción solicitada y qué vista debe mostrar como resultado.

```
<filter>
    <filter-name>struts2</filter-name>
    <filter-class>org.apache.struts2.dispatcher.FilterDispatcher</filter-
        class>
</filter>
<filter-mapping>
    <filter-name>struts2</filter-name>
    <url-pattern>/*</url-pattern>
</filter-mapping>
http://joeljil.wordpress.com/2010/05/31/struts2/
```

Listing 8.1: web.xml

El mapeado de una acción tiene la estructura mostrada en 8.2.

```
<action name=``Nombre_Accion" class=``Paquete.Nombre_Clase">
    <result name=``error">/pagina_error.jsp</result>
    <result name=``success">/pagina_exito.jsp</result>
</action>
```

Listing 8.2: web.xml

El *Nombre_Accion* es el nombre con el cual se llamará la acción desde la vista. *Paquete.Nombre_Clase* es la clase encargada de llevar a cabo la lógica requerida para que dicha acción se efectué. Por último los resultados *error* y *success* definen la vista que se presentará dependiendo del exito o fracaso de la acción.

La estructura anterior es la que siguen todas las acciones que hemos mapeado para nuestra aplicación.

Hemos organizado nuestro mapeado en diez archivos de mapeado. Cada uno de ellos agrupa las acciones de acuerdo a cada menú. En la figura 8.1 se muestra una imagen de los archivos de mapeo creados.

Archivo actividad.xml

Este archivo contiene el mapeo de las acciones relacionadas con la administración de actividades. Implica acciones tales como ver detalles de una actividad, agregar actividad, eliminar actividad, modificar una actividad. En la tabla 8.1 es muestran las acciones, clase, métodos y vistas implicados.

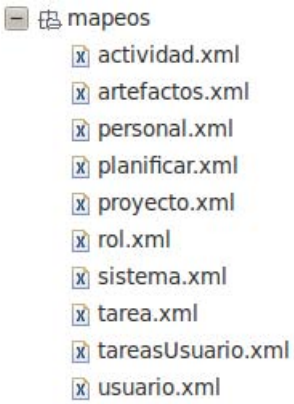

Figura 8.1: Organización de mapeados de acciones

Archivo artefactos.xml

En este archivo se incluyeron las acciones relacionadas con el manejo de los artefactos. Ver tabla 8.2.

Archivo personal.xml

En este archivo se han agrupado las acciones referentes con el manejo del personal de un proyecto. Ver tabla 8.3.

Archivo planificar.xml

Mapeo de accciones relacionadas con la planificación de un proyecto. Ver la tabla de la tabla 8.4.

Archivo proyecto.xml

En este archivo se mapearon las acciones que tienen que ver con la creación, eliminación , modificación y consultar información de un proyecto. Ver la tabla 8.5.

Archivo rol.xml

Acciones relacionadas con el submódulo de roles. Ver tabla 8.6.

Archivo sistema.xml

Mapeo de acciones comunes a toda la aplicación. Ver tabla 8.7.

Archivo tarea.xml

Mapeo de acciones necesarias para la administración de tareas en la planificación. Ver tabla 8.8.

Archivo tareasUsuario.xml

Mapeo de acciones para que el usuario administre las tareas que le han sido asignadas. Ver tabla 8.9.

Archivo usuario.xml

Acciones requeridas para la dmininistración de usuarios y permisos en el sistema. Ver tabla 8.10.

Acción	Clase/Método	Resultados	Tile
actividadesRup	ActividadAction, actividadesRup	succes, error	ActividadesRup, Error
detallesActividad	ActividadAction, detallesActividad	succes, error	DetallesActividad, Error
modificarEliminarActividad	ModificarEliminarActividadAction, modificarEliminarActividad	success, error	ModificarEliminarActividad, Error
guardarActividadModificada	ModificarEliminarActividadAction, guardarActividadModificada	success, error	ActividadModificada, Error
nuevaActividad	AgregarActividadAction, nuevaActividad	success, error	NuevaActividad, Error

Tabla 8.1: Tabla de acciones mapeadas en actividad.xml

Acción	Clase/Método	Resultados	Tile
artefactosSistema	ArtefactosSistemaAction, artefactosSistema	success, error	ArtefactosSistema, Error
artefactosProyecto	ArtefactosSistemaAction, artefactosProyecto	success, error	ArtefactosProyecto, Error
detallesArtefacto	ArtefactosSistemaAction, detallesArtefacto	success, error	DetallesArtefacto, Error
detallesRol	ArtefactosSistemaAction, detallesRol	success, error	DetallesRol, Error
nuevoArtefacto	AgregarArtefactoAction, nuevoArtefacto	success, error	NuevoArtefacto, Error
guardarArtefacto	AgregarArtefactoAction, guardarArtefacto	success, error	ArtefactoAgregado, Error
modificarEliminarArtefacto	ModificarEliminarArtefactoAction, modificarEliminarArtefacto	success, error	ModificarEliminarArtefacto, Error
guardarCambiosArtefacto	ModificarEliminarArtefactoAction, guardarCambiosArtefacto	success, error	ArtefactoModificado, Error
eliminarArtefacto	ModificarEliminarArtefactoAction, eliminarArtefacto	success, error	ArtefactoEliminado, Error

Tabla 8.2: Tabla de acciones mapeadas en artefactos.xml

Acción	Clase/Método	Resultados	Tile
personalProyecto	AdministrarPersonalAction, personalProyecto	login, error, success	proyectosUsuario, Error, PersonalProyecto
detallesUsuario	AdministrarPersonalAction, detallesUsuario	login, error, success	proyectosUsuario, Error, DetallesUsuario
detallesUsuario	AdministrarPersonalAction, detallesUsuario	login, error, success	proyectosUsuario, Error, DetallesUsuario
participantesProyecto	AgregarPersonalProyectoAction, participantesProyecto	login, error, success	proyectosUsuario, Error, AsignarParticipantesProyecto
asignarPersonalProyecto	AgregarPersonalProyectoAction, asignarPersonalProyecto	login, error, success	proyectosUsuario, Error, ListaPersonalActualizada
nuevoUsuario	NuevoPersonalAction, nuevoUsuario		NuevoUsuario
registrarUsuario	NuevoPersonalAction, registrarUsuario	input, login, error, success	NuevoUsuario, proyectosUsuario, Error, UsuarioRegistrado
modificarEliminarUsuario	ModificarEliminarUsuarioAction, modificarEliminarUsuario	login, error, success	proyectosUsuario, Error, ModificarEliminarUsuario
datosUsuario	ModificarEliminarUsuarioAction, datosUsuario	error, success	Error, ModificarEliminarUsuario
actualizarUsuario	ModificarEliminarUsuarioAction, actualizarUsuario	input, error, success	modificarEliminarUsuario, Error, UsuarioActualizado
eliminarParticipante	ModificarEliminarUsuarioAction, eliminarParticipante	input, error, success	modificarEliminarUsuario, Error, UsuarioEliminado

Tabla 8.3: Tabla de acciones mapeadas en personal.xml

Acción	**Clase/Método**	**Resultados**	**Tile**
iteraciones	IteracionAction, iteraciones	success, error	Iteraciones, proyectosUsuario
tareas	IteracionAction, tareas	success, error	Tareas, Error
nuevaIteracion	IteracionAction, nuevaIteracion	success, error	NuevaIteracion, Error
agregarIteracion	IteracionAction, agregarIteracion	success, error	Iteraciones, Error
eliminarIteracion	IteracionAction, eliminarIteracion	success, error	Iteraciones, Error
modificarEliminarIteracion	IteracionAction, modificarEliminarIteracion	success, error	ModificarEliminarIteracion, Error
guardarCambiosIteracion	IteracionAction, guardarCambiosIteracion	success, error	iteraciones, Error

Tabla 8.4: Tabla de acciones mapeadas en planificar.xml

Acción	**Clase/Método**	**Resultados**	**Tile**
proyectosUsuario	AdministrarProyectoAction, proyectosUsuario	success, login	ProyectosUsuario, Login
establecerProyectoActual	AdministrarProyectoAction, establecerProyectoActual	success, login, error	ProyectosUsuario, Login, /jsp/error/error.jsp
modificarEliminarProyecto	AdministrarProyectoAction, proyectosUsuario	success, error	ModificarEliminarProyecto, /jsp/error/error.jsp
datosProyectoModificarEliminar	AdministrarProyectoAction, datosProyecto	success, none, error	ModificarEliminarProyecto, proyectosUsuario, /jsp/error/error.jsp
detallesProyecto	AdministrarProyectoAction, detallesProyecto	success, none, error	DetallesProyecto, proyectosUsuario, /jsp/error/error.jsp
guardarProyecto	AdministrarProyectoAction, guardarProyecto	success, none, error	ProyectoModificadoEliminado, ModificarEliminarProyecto, /jsp/error/error.jsp
eliminarProyecto	AdministrarProyectoAction, eliminarProyecto	success, none, error	ProyectoModificadoEliminado, ModificarEliminarProyecto, /jsp/error/error.jsp
nuevoProyecto	AdministrarProyectoAction, verificarPermisos	success, none	NuevoProyecto, proyectosUsuario
registrarProyecto	AdministrarProyectoAction, registrarProyecto	success, none, error	ProyectoRegistrado, proyectosUsuario, /jsp/error/error.jsp

Tabla 8.5: Tabla de acciones mapeadas en proyecto.xml

Acción	**Clase/Método**	**Resultados**	**Tile**
rolesSistema	RolSistemaAction, rolesSistema	success, error	RolesSistema, Error
detallesRol2	RolSistemaAction, detallesRol	success, error	InfoRol, Error
nuevoRol		success, error	NuevoRol, Error
guardarRol	RolAction, guardarRol	success, error	RolAgregado, Error
modificarEliminarRol	RolAction, modificarEliminarRol	success, error	ModificarEliminarRol, Error
guardarCambiosRol	RolAction, guardarCambiosRol	success, error	InfoRol, Error
eliminarRol	RolAction, eliminarRol	success, error	InfoRol, Error

Tabla 8.6: Tabla de acciones mapeadas en rol.xml

Acción	**Clase/Método**	**Resultados**	**Tile**
inicio			Inicio
salir	ValidarUsuarioAction, terminarSesion	success, input	Login, Login
login		success	Login

Tabla 8.7: Tabla de acciones mapeadas en sistema.xml

Acción	**Clase/Método**	**Resultados**	**Tile**
nuevaTarea	NuevaTareaAction, nuevaTarea	success, error	NuevaTarea, Error
registrarTarea	NuevaTareaAction, guardarTarea	success, error	TareaRegistrada, Error
mostrarTareasIteracion	TareaAction, getJSON	json	Datos en JSON
detallesTarea	TareaAction, detallesTarea		DetallesTarea
modificarTarea	ModificarTareaAction, modificarTarea		ModificarTarea
guardarTareaModificada	ModificarTareaAction, guardarTareaModificada		TareaModificada
usuariosCandidatos	ModificarTareaAction, getJSONListaUsuariosCandidatos		Datos tipo JSON
actividadesDisciplina	NuevaTareaAction, getJSONActividadesDisciplina		Datos tipo JSON
subtareasDeTarea	NuevaTareaAction, getJSONsubtareas		Datos tipo JSON
eliminarTarea	EliminarTareaAction, eliminarTarea		TareaEliminada

Tabla 8.8: Tabla de acciones mapeadas en tarea.xml

Acción	**Clase/Método**	**Resultados**	**Tile**
tareasUsuario	TareasUsuarioAction, tareasUsuario	success, error	TareasUsuario, Error
detallesTareaUsuario	TareaUsuarioAction, detallesTareaUsuario	success, error	DetallesTareaUsuario, Error
verDetallesActividad	TareaUsuarioAction, verDetallesActividad	success, error	VerDetallesActividad, Error
verDetallesRol	TareaUsuarioAction, verDetallesRol	success, error	VerDetallesRol, Error
artefactosTarea	ArtefactosTareaAction, artefactosTarea	success, error	ArtefactosTarea, Error
seleccionarArchivo	SubirDescargarArtefactoAction, seleccionarArchivo	success, error	ElegirArchivo, Error
subirArchivoArtefacto	SubirDescargarArtefactoAction, subirEjemplar	success, error	SubidoExitosamente, Error
descargarArchivo	SubirDescargarArtefactoAction	success, error	Obtención del archivo, Error

Tabla 8.9: Tabla de acciones mapeadas en tareasUsuario.xml

Acción	**Clase/Método**	**Resultados**	**Tile**
listarPersonal			ListaPersonal
usuariosProyecto	AdministrarUsuariosAction, usuariosProyecto		AdministrarUsuarios
datosUsuario	AdministrarUsuariosAction, datosUsuario		AdministrarUsuarios
eliminarUsuario	AdministrarUsuariosAction, eliminarUsuario		UsuarioEliminado
guardarUsuario	AdministrarUsuariosAction, guardarUsuario		UsuarioActualizado
nuevoUsuario			NuevoUsuario
registrarUsuario	AdministrarUsuariosAction, guardarUsuario		UsuarioRegistrado
validarUsuario	ValidarUsuarioAction, validarUsuario	input, login, error, success	Login, Login, /jsp/error/error.jsp, proyectosUsuario
nuevaAccion		success	NuevaAccion
registrarAccion	SeguridadAction, registrarAccion	success	Permisos
modificarEliminarAccion	SeguridadAction, modificarEliminarAccion	success	ModificarEliminarAccion
eliminarAccion	SeguridadAction, eliminarAccion	success	Permisos
actualizarAccion	SeguridadAction, actualizarAccion	success	Permisos

Tabla 8.10: Tabla de acciones mapeadas en usuario.xml

Capítulo 9

Implementación del módulo de seguridad

9.1. Seguridad

El manejo de privilegios en toda aplicación es importante, con ello se evita que usuarios mal intencionados o usuarios con poco conocimiento del funcionamiento del sistema realicen acciones que resulten en pérdida de información, o simplemente para limitar privilegios a ciertos usuarios. En nuestro sistema hemos implementado de forma muy sencilla el manejo de privilegios. En las siguientes secciones describiremos las tereas que llevamos a cabo para implementar dicho módulo.

9.2. Base de Datos

Como primer paso para el inicio de este módulo se crearon dos tablas adicionales que contendrán la información suficiente para manejar los privilegios de acuerdo a cada rol. Debido a que cada petición que se invoca al servidor se conoce en Struts con el nombre de *Action*, hemos decidido nombrar a nuestra tabla como *Accion*. De tal forma que esta entidad almancena los siguietes datos: id de la acción, nombre de la acción, descripción de la acción y una categoría a la que pertenecerá la acción. El id de la acción será un autoincremental, el nombre corresponde con el nombre mapeado de la acción en los archivos XML que ya se comentaron anteriormente, la descripción es información descriptiva de lo que realiza la acción, y la categoría corresponde con uno de los submódulos que tenemos, tales como usuario, tareas, planificar, artefactos, actividades, entre otros.

Por otra parte, se identificó otra entidad que hemos nombrado como *Permiso*, que contiene la siguiente información. La acción, el rol, y un valor de si/no que nos indicará si dicha acción puede ser ejecutada por un usuario que tenga tal rol.

En la figura 9.1 se pueden ver las tablas creadas para el almacenamiento de la información necesaria para el manejo de permisos.

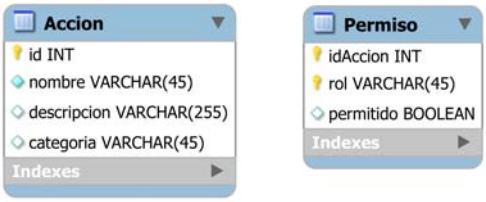

Figura 9.1: Tablas Accion y Permiso

Por otra parte las restricciones para la integridad referencial son:

```
 -- restricciones para la tabla Accion
PRIMARY KEY (‘id’),
UNIQUE KEY ‘nombreAccion’ (‘nombre’)


 -- restricciones para la tabla Permiso
PRIMARY KEY (‘idAccion’,‘rol’)
KEY ‘fk_roles’ (‘rol’),
CONSTRAINT ‘fk_accion’
FOREIGN KEY (‘idAccion’) REFERENCES ‘Accion’ (‘id’)
ON DELETE CASCADE ON UPDATE CASCADE,
CONSTRAINT ‘fk_roles’
FOREIGN KEY (‘rol’) REFERENCES ‘Rol‘ (‘nombreRol’)
```

9.3. Implementación de Clases

Para poder manejar las entidades mencionadas anteriormente hemos creado una clase persistente llamada *Accion*, la cual se observa parcialmente en 9.1.

En esta clase al igual que en las otras clases comentadas en secciones anteriores, hemos seguido con los estándares adoptados por lo que sólo ha sido necesario especificar el atributo que sirve como identificador de la clase.

También fue necesario crear la clase persistente llamada *Permiso*, la cual se observa en 9.2.

```
@Entity
public class Accion implements Serializable{
  private Integer id;
  private String nombre;
  private String descripcion;
  private String categoria;

  public Accion(){

  }

  @Id
  public Integer getId() {
    return id;
  }
```

Listing 9.1: Accion.java

Como se puede apreciar, esta clase tiene una asociación uno a uno con las clases *Accion* y *Rol*, la cual se evidencia por el mapeado realizado. Sin embargo, la tabla correspodiente a esta clase en la base de datos tiene una llave primaria compuesta por *idAccion,rol*, razón por la cual se ha creado una clase que representa dicha llave, esta se ve en 9.3, la cual hemos utilizado como identificador de la clase *Permiso.java*.

9.4. Implementación de vistas

Para poder administrar los permisos es necesario crear vistas para el usuario. Para este módulo hemos creado tres pantallas.

- Seguridad. Esta pantalla muestra en primer instancia las lista de los roles definidos en el sistema, en donde el usuario después de elegir un rol, consultará los permisos asociados a dicho rol. Desde esta pantalla el usuario podrá conceder o quitar permisos para el rol en cuentión. También desde esta pantalla podrá ir a una pantalla de modificación o eliminación de una acción. Ver figura 9.2.
- Nueva Accion. Esta pantalla permite al usuario agregar una nueva acción en el sistema. Ver figura 9.3.
- Modificar/Eliminar Accion. Desde esta pantalla se puede modificar información asociada a la acción seleccionada o bien eliminarla. Ver figura 9.4.

Con esto tenemos las pantallas necesarias para ofrecer la funcionalidad de manejo de privilegios en el sistema.

```
@Entity
public class Permiso implements Serializable{
  private IdPermiso id;
  private Accion accion;
  private Rol rol;
  private Boolean permitido;

  public Permiso(){

  }

  @EmbeddedId
  public IdPermiso getId() {
    return id;
  }
  public void setId(IdPermiso id) {
    this.id = id;
  }
  @OneToOne
  @JoinColumn(name="idAccion",insertable=false,updatable=false)
  public Accion getAccion() {
    return accion;
  }
  public void setAccion(Accion accion) {
    this.accion = accion;
  }
  @OneToOne
  @JoinColumn(name="rol",insertable=false,updatable=false)
  public Rol getRol() {
    return rol;
  }
```

Listing 9.2: Permiso.java

```java
@Embeddable
public class IdPermiso implements Serializable {
  private Integer idAccion;
  private String rol;

  public IdPermiso(){}

  public Integer getIdAccion() {
    return idAccion;
  }
  public void setIdAccion(Integer idAccion) {
    this.idAccion = idAccion;
  }
  public String getRol() {
    return rol;
  }
  public void setRol(String rol) {
    this.rol = rol;
  }
}
```

Listing 9.3: IdPermiso.java

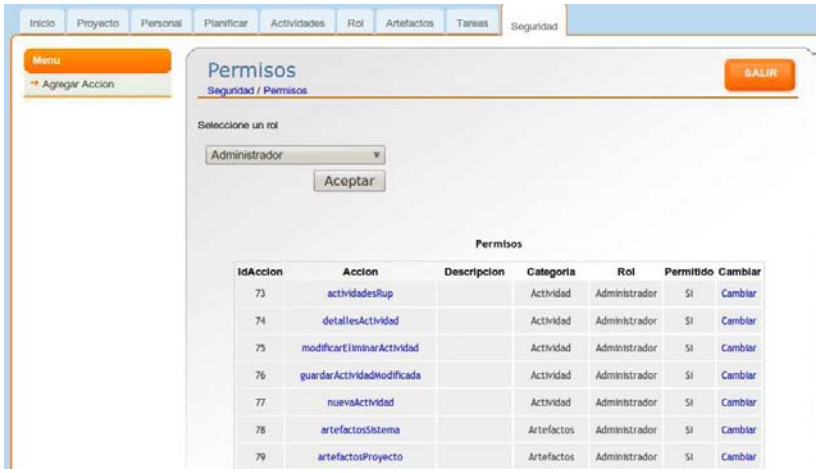

Figura 9.2: Lista de permisos de acuerdo al rol elegido

Figura 9.3: Pantalla para agregar una nueva acción en el sistema

Figura 9.4: Modificación o Eliminación de la acción

9.5. Validación de permisos

Ahora describiremos la lógica implementada para este módulo. Para la verificación de permisos seguiremos el flujo mostrado en la figura 9.5.

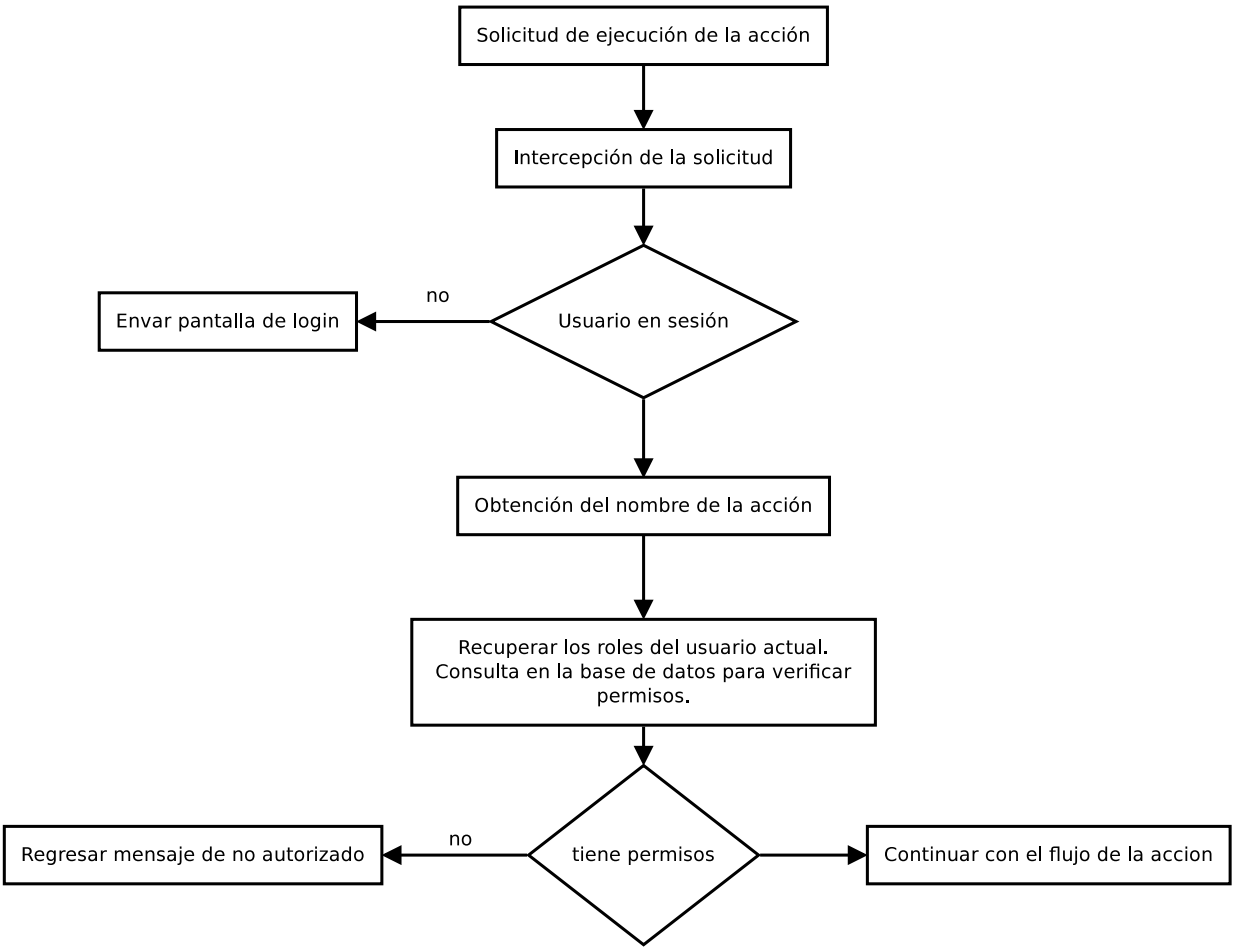

Figura 9.5: Diagrama de flujo para validación de permisos

La solicitud de la ejecución se da como resultado de un evento causado por el usuario, por ejemplo cuando hace click sobre un botón. Es entonces cuando se envía una petición al servidor. Sin embargo antes de que dicha petición se ejecute en el servidor, la interceptamos y validamos. Si el usuario en sesión tiene los permisos para ejecutarla permitimos que continue con su flujo normal, de lo contrario anulamos su flujo normal y enviamos un mensaje de no autorizado al usuario.

La parte medular de este módulo es la intercepción de la petición. Para ello hemos utilizado una herramienta que nos ofrece Struts, los interceptores. Un interceptor es código que se ejecuta antes y después de la ejecución de la acción solicitada. Mediante este código podemos alterar el flujo de la acción. Para poder crear un interceptor

basta con crear una clase que implemente la interface *Interceptor*, implementando los métodos *destroy, init e intercept*. El código del método *init* se ejecuta justo después de crearse el interceptor. El código del método *destroy* se ejecuta justo antes de destruirse el interceptor, y por último, el código del método *intercept*, es el método que llamará Struts.

Acorde a lo anterior, hemos creado el interceptor *ValidarPermisos*, el cual se ve en 9.4.

```
@Override
public String intercept(ActionInvocation action) throws Exception {
  ActionContext actionC = action.getInvocationContext();
  Map session = action.getInvocationContext().getSession();
   if (session.get("usuario") == null) {
        System.out.println("No esta logueado");
        return Action.LOGIN;
   } else {
        Usuario u = (Usuario)session.get("usuario");
        if(validarAccion(u.getRoles(),actionC.getName())){
        return action.invoke();
      }else{
        return "sinpermisos";
      }
      }
}

public boolean validarAccion(Set<Rol> roles, String accion){
  try{
    return manejadorUsuario.permitido(roles, accion);
  }catch (Exception e) {
    // TODO: handle exception
    e.printStackTrace();
  }
  return false;
}
```

Listing 9.4: ValidarPermisos.java

En la clase anterior se utiliza la clase *ActionContext* mediante la cual podemos obtener el nombre de la acción que ha sido invocada. Posteriormente recuperamos desde sesión el conjunto de roles del usuario. Posteriormente se verifican los permisos en el método *validarAccion*. Este método a su vez hace uso de la clase *ManejadorUsuario* en la cual hemos implementado un conjunto de métodos que nos auxilian en la consulta de datos. En particular el método *permitido* será muy usado, ya que cada acción se validará en la base de datos. Por lo anterior, la hemos implementado utilizando SQL en lugar de utilizar HQL. A continuación mostramos la consulta mencionada.

```
select p.idAccion, p.rol, p.permitido, a.nombre from Permiso as p
```

```
left outer join Accion as a  on p.idAccion = a.id
where p.rol in (‘ conjunto de roles del usuario’)
and a.nombre =‘accion’ and p.permitido=1;
```

Con lo ya descrito se tiene la implementación de las funcionalidades necesarias para el menejo de privilegios en el sistema, lo cual se puede realizar en la distintas pantallas mencionadas.

Capítulo 10

Conclusión

10.1. Conclusión

Con el paso de los años en el mundo del desarrollo de software se han ido descubriendo o creados métodos y técnicas para el desarrollo del software. El desarrollo de aplicaciones que sirvan de herramientas para dirigir proyectos de software ha ido creciendo también, el presente proyecto desarrollado es sólo un prototipo de una aplicación web que tiene por objetivo auxiliar en la dirección de proyectos de software basados en el proceso de desarrollo RUP. Debido a ello las funcionalidades ofrecidas se han adaptado a los requerimientos del proceso mencionado. Como habrá apreciado aún quedan muchos aspectos qué mejorar, los cuales se pueden realizar en otros proyectos o ir mejorando de forma gradual.

Mediante este proyecto desarrollado se ha adquirido cierta experiencia en diversos aspectos, tales como: la planeación de un proyecto, el análisis, el diseño y la implementación para obtener finalmente un producto funcional. Lo anterior ademas de permitir aplicar los conocimientos adquiridos a en la carrera de Ingeniería en Computación, nos ha permido ver las posibles mejoras para la misma aplicación. Ha permitido también constatar que si bién no se es experto en todas las etapas por las que se atraviesa al desarrollar un sistema software, ha dejado claro que se tienen los conocimientos necesarios para ofrecer una solución completa a problemas que requieren de un sistema de software.

Apéndice A

Códigos fuentes

```
<?xml version='1.0' encoding='utf-8'?>
<!DOCTYPE hibernate-configuration PUBLIC
        "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
        "http://hibernate.sourceforge.net/hibernate-configuration-3.0.
            dtd">

<hibernate-configuration>

  <session-factory>

    <!-- Database connection settings -->
    <property name="connection.driver_class">
      com.mysql.jdbc.Driver
    </property>
    <property name="connection.url">
      jdbc:mysql://localhost/prototipov6
    </property>
    <property name="connection.username">root</property>
    <property name="connection.password">root</property>

    <!-- JDBC connection pool (use the built-in) -->
    <property name="connection.pool_size">1</property>

    <!-- SQL dialect -->
    <property name="dialect">
      org.hibernate.dialect.MySQLDialect
    </property>

    <!-- Enable Hibernate's automatic session context management -->
    <property name="current_session_context_class">thread</property>
```

```
30
    <!-- Disable the second-level cache  -->
32  <property name="cache.provider_class">
      org.hibernate.cache.NoCacheProvider
34  </property>

36  <!-- Echo all executed SQL to stdout -->
    <property name="show_sql">false</property>
38
    <!-- Drop and re-create the database schema on startup -->
40  <property name="hbm2ddl.auto">update</property>


42
    <!-- <mapping resource="modelo/Usuario.hbm.xml"/>  -->
44  <mapping class="com.gestion.modelo.usuario.Usuario" />
    <mapping class="com.gestion.modelo.usuario.Accion" />
46  <mapping class="com.gestion.modelo.usuario.IdPermiso" />
    <mapping class="com.gestion.modelo.usuario.Permiso" />
48  <mapping class="com.gestion.modelo.rol.Rol" />
    <mapping class="com.gestion.modelo.proyecto.Proyecto" />
50  <mapping class="com.gestion.modelo.proyecto.Fase" />
    <mapping class="com.gestion.modelo.proyecto.Iteracion" />
52  <mapping class="com.gestion.modelo.proyecto.IdIteracion" />
    <mapping class="com.gestion.modelo.proyecto.Tarea" />
54  <mapping class="com.gestion.modelo.proyecto.Disciplina" />
    <mapping class="com.gestion.modelo.proyecto.Actividad" />
56  <mapping class="com.gestion.modelo.proyecto.Artefacto" />
    <mapping class="com.gestion.modelo.ejemplarArtefacto.
        EjemplarArtefacto"/>
58
  </session-factory>
60
</hibernate-configuration>
```

src/hibernate.cfg.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
2 <!DOCTYPE log4j:configuration PUBLIC "-//log4j/log4j Configuration//EN"
    "log4j.dtd">

4 <log4j:configuration xmlns:log4j="http://jakarta.apache.org/log4j/">

6     <!--
    <appender name="STDOUT" class="org.apache.log4j.ConsoleAppender">
8       <layout class="org.apache.log4j.PatternLayout">
          <param name="ConversionPattern" value="%d %-5p %c.%M:%L - %m%n
              "/>
10      </layout>
    </appender>
12 -->


14
    <!-- specify the logging level for loggers from other libraries -->
```

```
16
    <logger name="com.opensymphony">
18      <level value="DEBUG" />
    </logger>
20
    <logger name="org.apache.struts2">
22       <level value="DEBUG" />
    </logger>
24
   <!-- for all other loggers log only debug and above log messages -->
26
     <root>
28      <priority value="INFO"/>
        <appender-ref ref="STDOUT" />
30    </root>

32 </log4j:configuration>
```

src/log4j.xml

```
<?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
4   "http://struts.apache.org/dtds/struts-2.0.dtd">

6 <struts>
  <package name="tareasUsuario" extends="struts-default,json-default">
8   <result-types>
      <result-type name="tiles" class="org.apache.struts2.views.tiles.
          TilesResult" />
10  </result-types>

12  <interceptors>
      <interceptor name="authenticationInterceptor" class="com.
          interceptores.ValidarPermisos"/>
14    <interceptor-stack name="all-default">
        <interceptor-ref name="authenticationInterceptor" />
16      <interceptor-ref name="defaultStack"/>
      </interceptor-stack>
18  </interceptors>

20  <default-interceptor-ref name="all-default" />

22  <global-results>
      <result name="login" type="tiles">Login</result>
24    <result name="sinpermisos" type="tiles">SINPERMISOS</result>
    </global-results>
26
    <action name="tareasUsuario"  class="com.gestion.vista.tareasUsuario
        .TareasUsuarioAction"
28    method="tareasUsuario">
      <result name="success" type="tiles">TareasUsuario</result>
30    <result name="error" type="tiles">Error</result>
```

```
    </action>
    <action name="detallesTareaUsuario" class="com.gestion.vista.
        tareasUsuario.TareaUsuarioAction"
      method="detallesTareaUsuario">
      <result name="success" type="tiles">DetallesTareaUsuario</result>
      <result name="error" type="tiles">Error</result>
    </action>

    <action name="verDetallesActividad" class="com.gestion.vista.
        tareasUsuario.TareaUsuarioAction"
      method="verDetallesActividad">
      <result name="success" type="tiles">VerDetallesActividad</result>
      <result name="error" type="tiles">Error</result>
    </action>
    <action name="verDetallesRol" class="com.gestion.vista.
        tareasUsuario.TareaUsuarioAction"
      method="verDetallesRol">
      <result name="success" type="tiles">VerDetallesRol</result>
      <result name="error" type="tiles">Error</result>
    </action>
    <action name="artefactosTarea" class="com.gestion.vista.
        tareasUsuario.ArtefactosTareaAction"
      method="artefactosTarea">
      <result name="success" type="tiles">ArtefactosTarea</result>
      <result name="error" type="tiles">Error</result>
    </action>

    <action name="seleccionarArchivo" class="com.gestion.vista.
        tareasUsuario.SubirDescargarArtefactoAction"
      method="seleccionarArchivo">
      <result name="success" type="tiles">ElegirArchivo</result>
      <result name="error" type="tiles">Error</result>
    </action>

    <action name="subirArchivoArtefacto" class="com.gestion.vista.
        tareasUsuario.SubirDescargarArtefactoAction"
      method="subirEjemplar">
      <interceptor-ref name="fileUpload"/>
      <interceptor-ref name="basicStack"/>
      <result name="success" type="tiles">SubidoExitosamente</result>
      <result name="error" type="tiles">Error</result>
    </action>

    <action name="descargarArchivo" class="com.gestion.vista.
        tareasUsuario.SubirDescargarArtefactoAction">
        <result name="success" type="stream">
            <param name="inputName">inputStream</param>
            <param name="bufferSize">2048</param>
        </result>
        <result name="error" type="tiles">Error</result>
    </action>
  </package>
```

```
</struts>
```

src/mapeos/tareasUsuario.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
    "http://struts.apache.org/dtds/struts-2.0.dtd">

<struts>
  <package name="planificar" extends="struts-default">
    <result-types>
      <result-type name="tiles" class="org.apache.struts2.views.tiles.
          TilesResult" />
    </result-types>

    <interceptors>
        <interceptor name="authenticationInterceptor" class="com.
            interceptores.ValidarPermisos"/>
        <interceptor-stack name="all-default">
          <interceptor-ref name="authenticationInterceptor" />
          <interceptor-ref name="defaultStack"/>
        </interceptor-stack>
    </interceptors>

    <default-interceptor-ref name="all-default" />

    <global-results>
      <result name="login" type="tiles">Login</result>
      <result name="sinpermisos" type="tiles">SINPERMISOS</result>
    </global-results>

    <action name="iteraciones" class="com.gestion.vista.planificacion.
        IteracionAction"
      method="iteraciones">
      <result name="success" type="tiles">Iteraciones</result>
      <result name="error" type="redirectAction">proyectosUsuario</
          result>
    </action>

    <!--  haremos que este regrese un json -->
    <action name="tareas" class="com.gestion.vista.planificacion.
        IteracionAction"
      method="tareas">
      <result name="success" type="tiles">Tareas</result>
      <result name="error" type="tiles">Error</result>
    </action>

    <action name="nuevaIteracion" class="com.gestion.vista.planificacion
        .IteracionAction"
      method="nuevaIteracion">
      <result name="success" type="tiles">NuevaIteracion</result>
      <result name="error" type="tiles">Error</result>
```

```
44      </action>

46      <action name="agregarIteracion" class="com.gestion.vista.
            planificacion.IteracionAction"
          method="agregarIteracion">
48        <result name="success" type="tiles">Iteraciones</result>
          <result name="error" type="tiles">Error</result>
50      </action>

52      <action name="eliminarIteracion" class="com.gestion.vista.
            planificacion.IteracionAction"
          method="eliminarIteracion">
54        <result name="success" type="tiles">Iteraciones</result>
          <result name="error" type="tiles">Error</result>
56      </action>

58      <action name="modificarEliminarIteracion" class="com.gestion.vista.
            planificacion.IteracionAction"
          method="modificarEliminarIteracion">
60        <result name="success" type="tiles">ModificarEliminarIteracion</
              result>
          <result name="error" type="tiles">Error</result>
62      </action>

64      <action name="guardarCambiosIteracion" class="com.gestion.vista.
            planificacion.IteracionAction"
          method="guardarCambiosIteracion">
66        <result name="success" type="redirectAction">iteraciones</result>
          <result name="error" type="tiles">Error</result>
68      </action>
    </package>
70  </struts>
```

src/mapeos/planificar.xml

```
1   <?xml version="1.0" encoding="UTF-8"?>
    <!DOCTYPE struts PUBLIC
3       "-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
        "http://struts.apache.org/dtds/struts-2.0.dtd">
5
    <struts>
7     <package name="rol" extends="struts-default">
        <result-types>
9         <result-type name="tiles" class="org.apache.struts2.views.tiles.
              TilesResult" />
        </result-types>
11
        <interceptors>
13          <interceptor name="authenticationInterceptor" class="com.
                interceptores.ValidarPermisos"/>
            <interceptor-stack name="all-default">
15            <interceptor-ref name="authenticationInterceptor" />
              <interceptor-ref name="defaultStack"/>
```

```
17          </interceptor-stack>
    </interceptors>
19
    <default-interceptor-ref name="all-default" />
21
    <global-results>
23      <result name="login" type="tiles">Login</result>
        <result name="sinpermisos" type="tiles">SINPERMISOS</result>
25  </global-results>

27  <action name="rolesSistema" class="com.gestion.vista.rol.
        RolSistemaAction"
      method="rolesSistema">
29      <result name="success" type="tiles">RolesSistema</result>
        <result name="error" type="tiles">Error</result>
31  </action>

33  <action name="detallesRol2" class="com.gestion.vista.rol.
        RolSistemaAction"
      method="detallesRol">
35      <result name="success" type="tiles">InfoRol</result>
        <result name="error" type="tiles">Error</result>
37  </action>

39  <action name="nuevoRol">
        <result name="success" type="tiles">NuevoRol</result>
41      <result name="error" type="tiles">Error</result>
    </action>
43
    <action name="guardarRol" class="com.gestion.vista.rol.RolAction"
45      method="guardarRol">
        <result name="success" type="tiles">RolAgregado</result>
47      <result name="error" type="tiles">Error</result>
    </action>
49
    <action name="modificarEliminarRol" class="com.gestion.vista.rol.
        RolAction"
51      method="modificarEliminarRol">
        <result name="success" type="tiles">ModificarEliminarRol</result>
53      <result name="error" type="tiles">Error</result>
    </action>
55
    <action name="guardarCambiosRol" class="com.gestion.vista.rol.
        RolAction"
57      method="guardarCambiosRol">
        <result name="success" type="tiles">InfoRol</result>
59      <result name="error" type="tiles">Error</result>
    </action>
61
    <action name="eliminarRol" class="com.gestion.vista.rol.RolAction"
63      method="eliminarRol">
        <result name="success" type="tiles">InfoRol</result>
```

```
65        <result name="error" type="tiles">Error</result>
      </action>
67    <action name="pruebaEstilo">
        <result type="tiles">PruebaEstilo</result>
69    </action>
  </package>
71</struts>
```

src/mapeos/rol.xml

```
<?xml version="1.0" encoding="UTF-8"?>
2<!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
4   "http://struts.apache.org/dtds/struts-2.0.dtd">

6<struts>
  <package name="usuario" extends="struts-default">
8   <result-types>
      <result-type name="tiles" class="org.apache.struts2.views.tiles.
          TilesResult" />
10  </result-types>

12  <interceptors>
      <interceptor name="authenticationInterceptor" class="com.
          interceptores.ValidarPermisos"/>
14    <interceptor-stack name="all-default">
        <interceptor-ref name="authenticationInterceptor" />
16      <interceptor-ref name="defaultStack"/>
      </interceptor-stack>
18  </interceptors>

20  <default-interceptor-ref name="all-default" />

22  <global-results>
      <result name="login" type="tiles">Login</result>
24    <result name="sinpermisos" type="tiles">SINPERMISOS</result>
    </global-results>
26
    <action name="listarPersonal">
28    <result type="tiles">ListaPersonal</result>
    </action>
30  <!-- Desde aqui con diagramas -->
    <action name="usuariosProyecto" class="com.gestion.vista.usuarios.
        AdministrarUsuariosAction"
32      method="usuariosProyecto">
      <result type="tiles">AdministrarUsuarios</result>
34  </action>
    <action name="datosUsuario" class="com.gestion.vista.usuarios.
        AdministrarUsuariosAction"
36      method="datosUsuario">
      <result type="tiles">AdministrarUsuarios</result>
38  </action>
```

```
40      <action name="eliminarUsuario" class="com.gestion.vista.usuarios.
            AdministrarUsuariosAction"
          method="eliminarUsuario">
42        <result type="tiles">UsuarioEliminado</result>
        </action>
44
        <action name="guardarUsuario" class="com.gestion.vista.usuarios.
            AdministrarUsuariosAction"
          method="guardarUsuario">
46        <result type="tiles">UsuarioActualizado</result>
48      </action>

50      <action name="nuevoUsuario">
          <result type="tiles">NuevoUsuario</result>
52      </action>

54      <action name="registrarUsuario" class="com.gestion.vista.usuarios.
            AdministrarUsuariosAction"
          method="guardarUsuario">
56        <result type="tiles">UsuarioRegistrado</result>
        </action>
58
        <action name="verPermisos" class="com.gestion.vista.usuarios.
            SeguridadAction"
60        method="verPermisosDeRol">
          <result type="tiles">Permisos</result>
62      </action>

64      <action name="guardarCambiosPermisos" class="com.gestion.vista.
            usuarios.SeguridadAction"
          method="guardarCambiosPermisos">
66        <result name="success" type="tiles">Permisos</result>
        </action>
68
        <action name="nuevaAccion">
70        <result name="success" type="tiles">NuevaAccion</result>
        </action>
72      <action name="registrarAccion" class="com.gestion.vista.usuarios.
            SeguridadAction"
          method="registrarAccion">
74        <result name="success" type="tiles">Permisos</result>
        </action>
76      <action name="modificarEliminarAccion" class="com.gestion.vista.
            usuarios.SeguridadAction"
          method="modificarEliminarAccion">
78        <result name="success" type="tiles">ModificarEliminarAccion</
              result>
        </action>
80      <action name="eliminarAccion" class="com.gestion.vista.usuarios.
            SeguridadAction"
          method="eliminarAccion">
82        <result name="success" type="tiles">Permisos</result>
```

```
    </action>
84  <action name="actualizarAccion" class="com.gestion.vista.usuarios.
        SeguridadAction"
      method="actualizarAccion">
86    <result name="success" type="tiles">Permisos</result>
    </action>
88 </package>
</struts>
```

src/mapeos/usuario.xml

```
1 <?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE struts PUBLIC
3   "-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
    "http://struts.apache.org/dtds/struts-2.0.dtd">
5
<struts>
7 <package name="actividad" extends="struts-default">
    <result-types>
9     <result-type name="tiles" class="org.apache.struts2.views.tiles.
          TilesResult"/>
    </result-types>
11
    <interceptors>
13      <interceptor name="authenticationInterceptor" class="com.
            interceptores.ValidarPermisos"/>
        <interceptor-stack name="all-default">
15        <interceptor-ref name="authenticationInterceptor" />
          <interceptor-ref name="defaultStack"/>
17      </interceptor-stack>
    </interceptors>
19
    <default-interceptor-ref name="all-default" />
21
    <global-results>
23    <result name="login" type="tiles">Login</result>
      <result name="sinpermisos" type="tiles">SINPERMISOS</result>
25  </global-results>

27  <action name="actividadesRup" class="com.gestion.vista.actividad.
        ActividadAction"
      method="actividadesRup">
29    <result name="success" type="tiles">ActividadesRup</result>
      <result name="error" type="tiles">Error</result>
31  </action>

33  <action name="detallesActividad" class="com.gestion.vista.actividad.
        ActividadAction"
      method="detallesActividad">
35    <result name="success" type="tiles">DetallesActividad</result>
      <result name="error" type="tiles">Error</result>
37  </action>
```

```
39      <action name="modificarEliminarActividad" class="com.gestion.vista.
            actividad.ModificarEliminarActividadAction"
          method="modificarEliminarActividad">
41        <result name="success" type="tiles">ModificarEliminarActividad</
              result>
          <result name="error" type="tiles">Error</result>
43      </action>

45      <action name="guardarActividadModificada" class="com.gestion.vista.
            actividad.ModificarEliminarActividadAction"
          method="guardarActividadModificada">
47        <result name="success" type="tiles">ActividadModificada</result>
          <result name="error" type="tiles">Error</result>
49      </action>

51      <action name="nuevaActividad" class="com.gestion.vista.actividad.
            AgregarActividadAction"
          method="nuevaActividad">
53        <result name="success" type="tiles">NuevaActividad</result>
          <result name="error" type="tiles">Error</result>
55      </action>

57      <action name="agregarActividad" class="com.gestion.vista.actividad.
            AgregarActividadAction"
          method="agregarActividad">
59        <result name="success" type="tiles">ActividadAgregada</result>
          <result name="error" type="tiles">Error</result>
61      </action>
    </package>
63  </struts>
```

src/mapeos/actividad.xml

```
1   <?xml version="1.0" encoding="UTF-8"?>
    <!DOCTYPE struts PUBLIC
3       "-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
        "http://struts.apache.org/dtds/struts-2.0.dtd">
5
    <struts>
7     <package name="manejosession" extends="struts-default">
        <result-types>
9         <result-type name="tiles" class="org.apache.struts2.views.tiles.
              TilesResult" />
        </result-types>
11
        <action name="salirDelSistema" class="com.gestion.vista.usuarios.
            LogoutAction"
13        method="salirDelSistema">
          <result name="success" type="tiles">Login</result>
15      </action>

17      <action name="validarUsuario" class="com.gestion.vista.usuarios.
            ValidarUsuarioAction"
```

```
      method="validarUsuario">
        <result name="input" type="tiles">Login</result>
        <result name="login" type="tiles">Login</result>
        <result name="error" type="dispatcher">
            <param name="location">/jsp/error/error.jsp</param>
        </result>
        <result name="success" type="redirectAction">
            proyectosUsuario
        </result>
    </action>

    <action name="prueba" class="com.gestion.vista.ProbadorAction"
      method="prueba">
      <result>/jsp/Prueba.jsp</result>
    </action>
  </package>
</struts>
```

src/mapeos/session.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
    "http://struts.apache.org/dtds/struts-2.0.dtd">

<struts>
  <package name="tarea" extends="struts-default,json-default">
    <result-types>
      <result-type name="tiles" class="org.apache.struts2.views.tiles.
          TilesResult" />
    </result-types>

    <interceptors>
        <interceptor name="authenticationInterceptor" class="com.
            interceptores.ValidarPermisos"/>
        <interceptor-stack name="all-default">
          <interceptor-ref name="authenticationInterceptor" />
          <interceptor-ref name="defaultStack"/>
        </interceptor-stack>
    </interceptors>

    <default-interceptor-ref name="all-default" />

    <global-results>
      <result name="login" type="tiles">Login</result>
      <result name="sinpermisos" type="tiles">SINPERMISOS</result>
    </global-results>

    <action name="nuevaTarea"  class="com.gestion.vista.planificacion.
        NuevaTareaAction"
      method="nuevaTarea">
      <result name="success" type="tiles">NuevaTarea</result>
      <result name="error" type="tiles">Error</result>
```

```
    </action>
    <action name="registrarTarea"  class="com.gestion.vista.
        planificacion.NuevaTareaAction"
      method="guardarTarea">
      <result name="success" type="tiles">TareaRegistrada</result>
      <result name="error" type="tiles">Error</result>
    </action>
    <action name="tareasIteracion"  class="com.gestion.vista.
        planificacion.TareaAction"
      method="tareas">
      <result type="tiles">Inicio</result>
    </action>

    <action name="mostrarTareasIteracion" class="com.gestion.vista.
        planificacion.TareaAction"
    method="getJSON">
            <result type="json">
               <param name="excludeNullProperties">true</param>
            </result>
        </action>
        <action name="detallesTarea" class="com.gestion.vista.
            planificacion.TareaAction"
          method="detallesTarea">
            <result type="tiles">
                DetallesTarea
            </result>
        </action>

        <!-- Para modificar una tarea -->
        <action name="modificarTarea" class="com.gestion.vista.
            planificacion.ModificarTareaAction"
          method="modificarTarea">
            <result type="tiles">ModificarTarea</result>
        </action>
        <action name="guardarTareaModificada" class="com.gestion.vista.
            planificacion.ModificarTareaAction"
          method="guardarTareaModificada">
            <result type="tiles">TareaModificada</result>
        </action>

        <!-- Esta accion es solicitada por con ajax,
          para actualizar la lista de usuarios, una vez que se
             selecciono la actividad -->
        <action name="usuariosCandidatos" class="com.gestion.vista.
            planificacion.ModificarTareaAction"
    method="getJSONListaUsuariosCandidatos">
            <result type="json">
               <param name="excludeNullProperties">true</param>
            </result>
        </action>
```

```
74          <action name="actividadesDisciplina" class="com.gestion.vista.
                planificacion.NuevaTareaAction"
            method="getJSONActividadesDisciplina">
76              <result type="json">
                    <param name="excludeNullProperties">true</param>
78              </result>
            </action>
80
        <action name="subtareasDeTarea" class="com.gestion.vista.
            planificacion.NuevaTareaAction"
82          method="getJSONsubtareas">
                <result type="json">
84                  <param name="excludeNullProperties">true</param>
                </result>
86          </action>

88          <action name="eliminarTarea" class="com.gestion.vista.
                planificacion.EliminarTareaAction"
              method="eliminarTarea">
90              <result type="tiles">TareaEliminada</result>
            </action>
92    </package>
</struts>
```

src/mapeos/tarea.xml

```
1 <?xml version="1.0" encoding="UTF-8"?>
  <!DOCTYPE struts PUBLIC
3     "-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
      "http://struts.apache.org/dtds/struts-2.0.dtd">
5
  <struts>
7   <package name="administracion" extends="struts-default" >
      <result-types>
9       <result-type name="tiles" class="org.apache.struts2.views.tiles.
            TilesResult"/>
      </result-types>
11    <interceptors>
      <interceptor name="authenticationInterceptor"
13                    class="com.interceptores.InterceptorLogin"/>
        <interceptor-stack name="all-default">
15        <interceptor-ref name="authenticationInterceptor" />
          <interceptor-ref name="defaultStack"/>
17      </interceptor-stack>
    </interceptors>
19  <default-interceptor-ref name="all-default" />

21  </package>


23
    <package name="sistema" extends="struts-default">
25    <result-types>
```

```
      <result-type name="tiles" class="org.apache.struts2.views.tiles.
          TilesResult"/>
    </result-types>


    <action name="inicio">
      <result type="tiles">Inicio</result>
    </action>

    <action name="login">
      <result name="success" type="tiles">Login</result>
    </action>
  </package>
</struts>
```

src/mapeos/sistema.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
    "http://struts.apache.org/dtds/struts-2.0.dtd">

<struts>
  <package name="artefactos" extends="struts-default">
    <result-types>
      <result-type name="tiles" class="org.apache.struts2.views.tiles.
          TilesResult" />
    </result-types>

    <interceptors>
        <interceptor name="authenticationInterceptor" class="com.
            interceptores.ValidarPermisos"/>
        <interceptor-stack name="all-default">
          <interceptor-ref name="authenticationInterceptor" />
          <interceptor-ref name="defaultStack"/>
        </interceptor-stack>
    </interceptors>

    <default-interceptor-ref name="all-default" />

    <global-results>
      <result name="login" type="tiles">Login</result>
      <result name="sinpermisos" type="tiles">SINPERMISOS</result>
    </global-results>


    <action name="artefactosSistema" class="com.gestion.vista.artefactos
        .ArtefactosSistemaAction"
      method="artefactosSistema">
      <result name="success" type="tiles">ArtefactosSistema</result>
      <result name="error" type="tiles">Error</result>
    </action>
```

```
34      <action name="artefactosProyecto" class="com.gestion.vista.
            artefactos.ArtefactosSistemaAction"
          method="artefactosProyecto">
36        <result name="success" type="tiles">ArtefactosProyecto</result>
          <result name="error" type="tiles">Error</result>
38      </action>

40

        <action name="detallesArtefacto" class="com.gestion.vista.artefactos
            .ArtefactosSistemaAction"
42        method="detallesArtefacto">
          <result name="success" type="tiles">DetallesArtefacto</result>
44        <result name="error" type="tiles">Error</result>
        </action>
46
        <action name="detallesRol" class="com.gestion.vista.artefactos.
            ArtefactosSistemaAction"
48        method="detallesRol">
          <result name="success" type="tiles">DetallesRol</result>
50        <result name="error" type="tiles">Error</result>
        </action>
52
        <action name="nuevoArtefacto" class="com.gestion.vista.artefactos.
            AgregarArtefactoAction"
54        method="nuevoArtefacto">
          <result name="success" type="tiles">NuevoArtefacto</result>
56        <result name="error" type="tiles">Error</result>
        </action>
58

60      <action name="guardarArtefacto" class="com.gestion.vista.artefactos.
            AgregarArtefactoAction"
          method="guardarArtefacto">
62        <result name="success" type="tiles">ArtefactoAgregado</result>
          <result name="error" type="tiles">Error</result>
64      </action>

66      <action name="modificarEliminarArtefacto" class="com.gestion.vista.
            artefactos.ModificarEliminarArtefactoAction"
          method="modificarEliminarArtefacto">
68        <result name="success" type="tiles">ModificarEliminarArtefacto</
              result>
          <result name="error" type="tiles">Error</result>
70      </action>

72      <action name="guardarCambiosArtefacto" class="com.gestion.vista.
            artefactos.ModificarEliminarArtefactoAction"
          method="guardarCambiosArtefacto">
74        <result name="success" type="tiles">ArtefactoModificado</result>
          <result name="error" type="tiles">Error</result>
76      </action>
```

```
78      <action name="eliminarArtefacto" class="com.gestion.vista.artefactos
            .ModificarEliminarArtefactoAction"
          method="eliminarArtefacto">
80        <result name="success" type="tiles">ArtefactoEliminado</result>
          <result name="error" type="tiles">Error</result>
82      </action>
    </package>
84  </struts>
```

src/mapeos/artefactos.xml

```
1   <?xml version="1.0" encoding="UTF-8"?>
    <!DOCTYPE struts PUBLIC
3       "-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
        "http://struts.apache.org/dtds/struts-2.0.dtd">
5
    <struts>
7     <package name="proyecto" extends="struts-default">
        <result-types>
9       <result-type name="tiles" class="org.apache.struts2.views.tiles.
            TilesResult"/>
      </result-types>
11
      <interceptors>
13      <interceptor name="authenticationInterceptor" class="com.
            interceptores.ValidarPermisos"/>
        <interceptor-stack name="all-default">
15        <interceptor-ref name="authenticationInterceptor" />
          <interceptor-ref name="defaultStack"/>
17      </interceptor-stack>
      </interceptors>
19
      <default-interceptor-ref name="all-default" />
21
      <global-results>
23      <result name="login" type="tiles">Login</result>
        <result name="sinpermisos" type="tiles">SINPERMISOS</result>
25    </global-results>

27
      <action name="proyectosUsuario" class="com.gestion.vista.proyecto.
          AdministrarProyectoAction"
29      method="proyectosUsuario">
        <result name="success" type="tiles">ProyectosUsuario</result>
31      <result name="login" type="tiles">Login</result>
      </action>
33
      <action name="establecerProyectoActual" class="com.gestion.vista.
          proyecto.AdministrarProyectoAction"
35      method="establecerProyectoActual">
        <result name="success" type="tiles">ProyectosUsuario</result>
37      <result name="login" type="tiles">Login</result>
        <result name="error" type="dispatcher">
```

```
39              <param name="location">/jsp/error/error.jsp</param>
        </result>
41      </action>
        <action name="modificarEliminarProyecto" class="com.gestion.vista.
            proyecto.AdministrarProyectoAction"
43          method="proyectosUsuario">
            <result name="success" type="tiles">ModificarEliminarProyecto</
                result>
45          <result name="error" type="dispatcher">
                <param name="location">/jsp/error/error.jsp</param>
47          </result>
        </action>
49
        <action name="datosProyectoModificarEliminar" class="com.gestion.
            vista.proyecto.AdministrarProyectoAction"
51          method="datosProyecto">
            <result name="success" type="tiles">ModificarEliminarProyecto</
                result>
53          <result name="none" type="redirectAction">proyectosUsuario</result
                >
            <result name="error" type="dispatcher">
55              <param name="location">/jsp/error/error.jsp</param>
            </result>
57      </action>

59      <action name="detallesProyecto" class="com.gestion.vista.proyecto.
            AdministrarProyectoAction"
            method="detallesProyecto">
61          <result name="success" type="tiles">DetallesProyecto</result>
            <result name="none" type="redirectAction">proyectosUsuario</result
                >
63          <result name="error" type="dispatcher">
                <param name="location">/jsp/error/error.jsp</param>
65          </result>
        </action>
67

69      <action name="guardarProyecto" class="com.gestion.vista.proyecto.
            AdministrarProyectoAction"
            method="guardarProyecto">
71          <result name="success" type="tiles">ProyectoModificadoEliminado</
                result>
            <result name="none" type="tiles">ModificarEliminarProyecto</result
                >
73          <result name="error" type="dispatcher">
                <param name="location">/jsp/error/error.jsp</param>
75          </result>
        </action>
77
        <action name="eliminarProyecto" class="com.gestion.vista.proyecto.
            AdministrarProyectoAction"
79          method="eliminarProyecto">
```

```
      <result name="success" type="tiles">ProyectoModificadoEliminado</
          result>
      <result name="none" type="tiles">ModificarEliminarProyecto</result
          >
      <result name="error" type="dispatcher">
            <param name="location">/jsp/error/error.jsp</param>
      </result>
    </action>

    <action name="nuevoProyecto" class="com.gestion.vista.proyecto.
        AdministrarProyectoAction"
      method="verificarPermisos">
      <result name="success" type="tiles">NuevoProyecto</result>
      <result name="none" type="redirectAction">proyectosUsuario</result
          >
    </action>

    <action name="registrarProyecto" class="com.gestion.vista.proyecto.
        AdministrarProyectoAction"
      method="registrarProyecto">
      <result name="success" type="tiles">ProyectoRegistrado</result>
      <result name="none" type="redirectAction">proyectosUsuario</result
          >
      <result name="error" type="dispatcher">
            <param name="location">/jsp/error/error.jsp</param>
      </result>
    </action>


  </package>
</struts>
```

src/mapeos/proyecto.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
    "http://struts.apache.org/dtds/struts-2.0.dtd">

<struts>
  <package name="personal" extends="struts-default">
    <result-types>
      <result-type name="tiles" class="org.apache.struts2.views.tiles.
          TilesResult" />
    </result-types>

    <interceptors>
        <interceptor name="authenticationInterceptor" class="com.
            interceptores.ValidarPermisos"/>
        <interceptor-stack name="all-default">
          <interceptor-ref name="authenticationInterceptor" />
          <interceptor-ref name="defaultStack"/>
        </interceptor-stack>
```

```
18      </interceptors>

20      <default-interceptor-ref name="all-default" />

22      <global-results>
          <result name="login" type="tiles">Login</result>
24        <result name="sinpermisos" type="tiles">SINPERMISOS</result>
        </global-results>
26
        <action name="personalProyecto" class="com.gestion.vista.personal.
            AdministrarPersonalAction"
28        method="personalProyecto">
          <result name="login" type="redirectAction">proyectosUsuario</
              result>
30        <result name="error" type="tiles">Error</result>
          <result name="success" type="tiles">PersonalProyecto</result>
32      </action>

34      <action name="detallesUsuario" class="com.gestion.vista.personal.
            AdministrarPersonalAction"
          method="detallesUsuario">
36        <result name="login" type="redirectAction">proyectosUsuario</
              result>
          <result name="error" type="tiles">Error</result>
38        <result name="success" type="tiles">DetallesUsuario</result>
        </action>
40
        <action name="participantesProyecto" class="com.gestion.vista.
            personal.AgregarPersonalProyectoAction"
42        method="participantesProyecto">
          <result name="login" type="redirectAction">proyectosUsuario</
              result>
44        <result name="error" type="tiles">Error</result>
          <result name="success" type="tiles">AsignarParticipantesProyecto</
              result>
46      </action>

48
        <action name="asignarPersonalProyecto" class="com.gestion.vista.
            personal.AgregarPersonalProyectoAction"
50        method="asignarPersonalProyecto">
          <result name="login" type="redirectAction">proyectosUsuario</
              result>
52        <result name="error" type="tiles">Error</result>
          <result name="success" type="tiles">ListaPersonalActualizada</
              result>
54      </action>

56      <action name="nuevoUsuario" class="com.gestion.vista.personal.
            NuevoPersonalAction"
          method="nuevoUsuario">
58        <result type="tiles">NuevoUsuario</result>
```

```
    </action>

    <action name="registrarUsuario" class="com.gestion.vista.personal.
        NuevoPersonalAction"
      method="registrarUsuario">
      <result name="input" type="tiles">NuevoUsuario</result>
      <result name="login" type="redirectAction">proyectosUsuario</
          result>
      <result name="error" type="tiles">Error</result>
      <result name="success" type="tiles">UsuarioRegistrado</result>
    </action>

    <action name="modificarEliminarUsuario" class="com.gestion.vista.
        personal.ModificarEliminarUsuarioAction"
    method="modificarEliminarUsuario">
      <result name="login" type="redirectAction">proyectosUsuario</
          result>
      <result name="error" type="tiles">Error</result>
      <result name="success" type="tiles">ModificarEliminarUsuario</
          result>
    </action>

    <action name="datosUsuario" class="com.gestion.vista.personal.
        ModificarEliminarUsuarioAction"
    method="datosUsuario">
      <result name="error" type="tiles">Error</result>
      <result name="success" type="tiles">ModificarEliminarUsuario</
          result>
    </action>

    <action name="actualizarUsuario" class="com.gestion.vista.personal.
        ModificarEliminarUsuarioAction"
    method="actualizarUsuario">
      <result name="input" type="redirectAction">
          modificarEliminarUsuario</result>
      <result name="error" type="tiles">Error</result>
      <result name="success" type="tiles">UsuarioActualizado</result>
    </action>

    <action name="eliminarParticipante" class="com.gestion.vista.
        personal.ModificarEliminarUsuarioAction"
    method="eliminarParticipante">
      <result name="input" type="redirectAction">
          modificarEliminarUsuario</result>
      <result name="error" type="tiles">Error</result>
      <result name="success" type="tiles">UsuarioEliminado</result>
    </action>
  </package>
</struts>
```

src/mapeos/personal.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
    "http://struts.apache.org/dtds/struts-2.0.dtd">

<struts>
  <include file="mapeos/sistema.xml"></include>
  <include file="mapeos/usuario.xml"></include>
  <include file="mapeos/proyecto.xml"></include>
  <include file="mapeos/rol.xml"></include>
  <include file="mapeos/planificar.xml"></include>
  <include file="mapeos/tarea.xml"></include>
  <include file="mapeos/personal.xml"></include>
  <include file="mapeos/artefactos.xml"></include>
  <include file="mapeos/actividad.xml"></include>
  <include file="mapeos/tareasUsuario.xml"></include>
  <include file="mapeos/session.xml"></include>
</struts>
```

src/struts.xml

```
package com.gestion.modelo.plantilla;

import java.io.Serializable;
import java.sql.Blob;
import java.sql.Date;

import javax.persistence.EmbeddedId;
import javax.persistence.JoinColumn;
import javax.persistence.JoinColumns;
import javax.persistence.OneToOne;

import com.gestion.modelo.proyecto.Artefacto;
import com.gestion.modelo.proyecto.Proyecto;

public class Plantilla implements Serializable {
  private IdPlantilla id;
  private String nombreArchivo;
  private String tipoContenido;
  private Blob contenido;
  private Date fechaCreacion;
  private Artefacto artefacto;
  private Proyecto proyecto;
  @EmbeddedId
  public IdPlantilla getId() {
    return id;
  }
  public void setId(IdPlantilla id) {
    this.id = id;
  }
  public String getNombreArchivo() {
    return nombreArchivo;
  }
```

```
    public void setNombreArchivo(String nombreArchivo) {
34    this.nombreArchivo = nombreArchivo;
    }
36  public String getTipoContenido() {
      return tipoContenido;
38  }
    public void setTipoContenido(String tipoContenido) {
40    this.tipoContenido = tipoContenido;
    }
42  public Blob getContenido() {
      return contenido;
44  }
    public void setContenido(Blob contenido) {
46    this.contenido = contenido;
    }
48  public Date getFechaCreacion() {
      return fechaCreacion;
50  }
    public void setFechaCreacion(Date fechaCreacion) {
52    this.fechaCreacion = fechaCreacion;
    }
54  @OneToOne
    @JoinColumn(name="Artefacto_nombre",insertable=false, updatable=false)
56  public Artefacto getArtefacto() {
      return artefacto;
58  }
    public void setArtefacto(Artefacto artefacto) {
60    this.artefacto = artefacto;
    }
62  @OneToOne
    @JoinColumn(name="Proyecto_idProyecto", insertable=false, updatable=
        false)
64  public Proyecto getProyecto() {
      return proyecto;
66  }
    public void setProyecto(Proyecto proyecto) {
68    this.proyecto = proyecto;
    }
70 }
```

src/com/gestion/modelo/plantilla/Plantilla.java

```
  package com.gestion.modelo.plantilla;
2
  import java.io.Serializable;
4
  import javax.persistence.Column;
6
  public class IdPlantilla implements Serializable{
8   private long numProyecto;
    private String nombreArtefacto;
10
    @Column(name="Proyecto_idProyecto")
```

```
12    public long getNumProyecto() {
        return numProyecto;
14    }
      public void setNumProyecto(long numProyecto) {
16      this.numProyecto = numProyecto;
      }
18    @Column(name="Artefacto_nombre")
      public String getNombreArtefacto() {
20      return nombreArtefacto;
      }
22    public void setNombreArtefacto(String nombreArtefacto) {
        this.nombreArtefacto = nombreArtefacto;
24    }
}
```

src/com/gestion/modelo/plantilla/IdPlantilla.java

```
1  package com.gestion.modelo.proyecto;

3  import java.io.Serializable;

5  import javax.persistence.Entity;
   import javax.persistence.Id;
7
   @Entity
9  public class Fase implements Serializable{
     private long idFase;
11   private String nombre;
     private String descripcion;
13
     public Fase(){
15
     }
17   @Id
     public long getIdFase() {
19     return idFase;
     }
21
     public void setIdFase(long idFase) {
23     this.idFase = idFase;
     }
25
     public String getNombre() {
27     return nombre;
     }
29
     public void setNombre(String nombre) {
31     this.nombre = nombre;
     }
33
     public String getDescripcion() {
35     return descripcion;
     }
```

```
37
  public void setDescripcion(String descripcion) {
39   this.descripcion = descripcion;
  }
41 }
```

src/com/gestion/modelo/proyecto/Fase.java

```
1  package com.gestion.modelo.proyecto;

3  import java.io.IOException;
   import java.io.InputStream;
5  import java.io.Serializable;
   import java.sql.Blob;
7  import java.sql.Date;
   import java.sql.SQLException;
9
   import javax.persistence.Entity;
11 import javax.persistence.Id;
   import javax.persistence.JoinColumn;
13 import javax.persistence.Lob;
   import javax.persistence.OneToOne;
15
   import org.hibernate.Hibernate;
17
   import com.gestion.modelo.rol.Rol;
19
   @Entity
21 public class Artefacto implements Serializable{
     private String nombre;
23   private String descripcion;
     private String urlDescripcion;
25   private Rol rol;

27   //Agregados el 30 de junio de 2010
     private String proposito;
29   private String timing;
     private String tailoring;
31   private String representacionUML;

33   public Artefacto(){

35   }
     @Id
37   public String getNombre() {
       return nombre;
39   }

41   public void setNombre(String nombre) {
       this.nombre = nombre;
43   }

45   public String getDescripcion() {
```

```
    return descripcion;
  }

  public void setDescripcion(String descripcion) {
    this.descripcion = descripcion;
  }

  public String getUrlDescripcion() {
    return urlDescripcion;
  }

  public void setUrlDescripcion(String urlDescripcion) {
    this.urlDescripcion = urlDescripcion;
  }


  @OneToOne
  @JoinColumn(name="Rol_nombreRol")
  public Rol getRol() {
    return rol;
  }
  public void setRol(Rol rol) {
    this.rol = rol;
  }
  public String getProposito() {
    return proposito;
  }
  public void setProposito(String proposito) {
    this.proposito = proposito;
  }
  public String getTiming() {
    return timing;
  }
  public void setTiming(String timing) {
    this.timing = timing;
  }
  public String getTailoring() {
    return tailoring;
  }
  public void setTailoring(String tailoring) {
    this.tailoring = tailoring;
  }
  public String getRepresentacionUML() {
    return representacionUML;
  }
  public void setRepresentacionUML(String representacionUML) {
    this.representacionUML = representacionUML;
  }
}
```

src/com/gestion/modelo/proyecto/Artefacto.java

```
package com.gestion.modelo.proyecto;

public enum NombreFase{Inicio,Elaboracion,Construccion,Transicion}
```

src/com/gestion/modelo/proyecto/NombreFase.java

```
package com.gestion.modelo.proyecto;

import java.io.Serializable;

import javax.persistence.Column;
import javax.persistence.Embeddable;

@Embeddable
public class IdIteracion implements Serializable{
  private long numIteracion;
  private long idProyecto;

  public IdIteracion(){

  }

  public IdIteracion(long idProyecto, long numIteracion){
    this.idProyecto = idProyecto;
    this.numIteracion = numIteracion;
  }

  public long getNumIteracion() {
    return numIteracion;
  }
  public void setNumIteracion(long numIteracion) {
    this.numIteracion = numIteracion;
  }
  @Column(name="Proyecto_idProyecto")
  public long getIdProyecto() {
    return idProyecto;
  }
  public void setIdProyecto(long idProyecto) {
    this.idProyecto = idProyecto;
  }
}
```

src/com/gestion/modelo/proyecto/IdIteracion.java

```
package com.gestion.modelo.proyecto;

import java.io.Serializable;
import java.sql.Date;
import java.util.List;

import javax.persistence.Column;
import javax.persistence.EmbeddedId;
import javax.persistence.Entity;
```

```
import javax.persistence.FetchType;
import javax.persistence.JoinColumn;
import javax.persistence.JoinColumns;
import javax.persistence.JoinTable;
import javax.persistence.ManyToOne;
import javax.persistence.OneToMany;
import javax.persistence.OneToOne;

import org.hibernate.annotations.Columns;

@Entity
public class Iteracion implements Serializable{
  private IdIteracion id;
  private String nombre;
  private String descripcion;
  private Date fechaInicio;
  private long indice;
  private Fase fase;
  private Proyecto proyecto;
  /**
   * Una Iteracion esta conformada de muchas tareas
   */
  private List<Tarea> tareas;

  public Iteracion(){

  }
  @EmbeddedId
  public IdIteracion getId() {
    return id;
  }

  public void setId(IdIteracion id) {
    this.id = id;
  }
  public String getNombre() {
    return nombre;
  }
  public void setNombre(String nombre) {
    this.nombre = nombre;
  }
  public String getDescripcion() {
    return descripcion;
  }

  public void setDescripcion(String descripcion) {
    this.descripcion = descripcion;
  }

  public Date getFechaInicio() {
    return fechaInicio;
  }
```

```
63    public void setFechaInicio(Date fechaInicio) {
        this.fechaInicio = fechaInicio;
65    }

67    public long getIndice() {
        return indice;
69    }

71    public void setIndice(long indice) {
        this.indice = indice;
73    }
      /**
75     * Una iteracion esta asociada con una fase
       * Para saber que fase es basta consultar la columna 'Fase_idFase'
77     * Con EAGER al momento de recuperar la Iteracion, carga la Fase
       * @return
79     */
      @OneToOne
81    @JoinColumn(name="Fase_idFase")
      public Fase getFase() {
83      return fase;
      }
85    public void setFase(Fase fase) {
        this.fase = fase;
87    }
      /**
89     * Esta es una asociacion uno a muchos, en donde se utiliza una
       * tabla para la asociacion. Una iteracion tiene muchas tareas,
91     * para averiguarlo utilice la tabla  'IteracionTareas' y todos
       * los que tengan el identificador de esta iteracion, es decir 'id'
93     * corresponden a las tareas de esta iteracion.
       * @return
95     */
      //@Column(name="Iteracion_numIteracion"),
97    //@Column(name="Iteracion_Proyecto_idProyecto")
      @OneToMany(mappedBy="iteracion")
99    @Columns(columns = { @Column(name="Iteracion_numIteracion"),@Column(
          name="Iteracion_Proyecto_idProyecto") })
      public List<Tarea> getTareas() {
101     return tareas;
      }
103   public void setTareas(List<Tarea> tareas) {
        this.tareas = tareas;
105   }
      /**
107    * Varias entidades Iteracion estas asociadas con un proyecto, para
           saber cual es
       * basta consultar la columna 'Proyecto_idProyecto'
109    * Al recuperar la iteracion recupera el Proyecto al que pertenece
       * @return
111    */
```

```
  @ManyToOne
  @JoinColumn(name="Proyecto_idProyecto",insertable=false, updatable=
      false)
//  @JoinColumn(name="Proyecto_idProyecto")
  public Proyecto getProyecto() {
    return proyecto;
  }
  public void setProyecto(Proyecto proyecto) {
    this.proyecto = proyecto;
  }


}
```

src/com/gestion/modelo/proyecto/Iteracion.java

```
package com.gestion.modelo.proyecto;

import java.io.Serializable;
import java.sql.Date;
import java.util.ArrayList;
import java.util.HashSet;
import java.util.Iterator;
import java.util.List;
import java.util.Set;

import javax.persistence.Column;
import javax.persistence.EmbeddedId;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.JoinColumns;
import javax.persistence.JoinTable;
import javax.persistence.ManyToMany;
import javax.persistence.ManyToOne;
import javax.persistence.OneToMany;
import javax.persistence.OneToOne;
import javax.persistence.Transient;

import com.gestion.modelo.usuario.Usuario;

@Entity
public class Tarea implements Serializable {
  /**
   *
   */
  private static final long serialVersionUID = 1L;
  private IdTarea idTarea;
  private String nombreTarea;
  private Date fechaInicio;
  private Date fechaFin;
  private int hito;
```

```
    private int porcentaje;
39  private int pGroup;
    //private Tarea tareaPadre;
41  /**
     * Cardinalidad 0 o 1
43   */
    private List<Tarea> subtareas = new ArrayList<Tarea>();
45  private List<Tarea> tareaPadre = new ArrayList<Tarea>();
    private int pOpen;
47  private String pCaption;
    private int indice;
49  private Iteracion iteracion;
    private List<Tarea> predecesores = new ArrayList<Tarea>();
51  private Actividad actividad;
    private Set<Usuario> recursos = new HashSet<Usuario>();
53
    public Tarea() {
55    // TODO Auto-generated constructor stub
    }
57  @EmbeddedId
    public IdTarea getIdTarea() {
59    return idTarea;
    }
61
    public void setIdTarea(IdTarea idTarea) {
63    this.idTarea = idTarea;
    }
65
    public String getNombreTarea() {
67    return nombreTarea;
    }
69
    public void setNombreTarea(String nombreTarea) {
71    this.nombreTarea = nombreTarea;
    }
73  //@Temporal(TemporalType.DATE)
    public Date getFechaInicio() {
75    return fechaInicio;
    }
77  public void setFechaInicio(Date fechaInicio) {
      this.fechaInicio = fechaInicio;
79  }
    //@Temporal(TemporalType.DATE)
81  public Date getFechaFin() {
      return fechaFin;
83  }
    public void setFechaFin(Date fechaFin) {
85    this.fechaFin = fechaFin;
    }
87


89  public int getPorcentaje() {
```

```
    return porcentaje;
91  }
    public void setPorcentaje(int porcentaje) {
93    this.porcentaje = porcentaje;
    }
95  public int getIndice() {
      return indice;
97  }

99  public void setIndice(int indice) {
      this.indice = indice;
101 }

103 /**
     * Muchas Tareas estan asociadas a una Iteracion, para
105  * saber que Iteracion es basta con consultar las columnas
     * 'Iteracion_Proyecto_idProyecto' e 'Iteracion_numIteracion'
107  * @return
     */
109 @ManyToOne
    @JoinColumns({
111     @JoinColumn(name="Iteracion_numIteracion", referencedColumnName="
            numIteracion",insertable=false,updatable=false),
        @JoinColumn(name="Iteracion_Proyecto_idProyecto",
            referencedColumnName="Proyecto_idProyecto",insertable=false,
            updatable=false)
113 })
    public Iteracion getIteracion() {
115   return iteracion;
    }
117 public void setIteracion(Iteracion iteracion) {
      this.iteracion = iteracion;
119 }
    public int getHito() {
121   return hito;
    }
123 public void setHito(int hito) {
      this.hito = hito;
125 }
    @Column(name="pGroup")
127 public int getpGroup() {
      return pGroup;
129 }
    public void setpGroup(int pGroup) {
131   this.pGroup = pGroup;
    }
133 /**
     * Esta es una sociacion uno a uno unidireccional
135  * Lo anterior es incorrecto, una tarea puede tener muchas subtareas
     * en la columna 'pParent' esta el identificador
137  * de la tarea padre, lo recupere al momento de recuperar la istancia
         (EAGER)
```

```
     * @return
139  */
    @OneToMany(fetch=FetchType.EAGER)
141 @JoinTable(name="PadreTareaHijo",
        joinColumns={
143     @JoinColumn(name="numTareaHijo",referencedColumnName="numTarea"),
        @JoinColumn(name="numProyectoH",referencedColumnName="
            Iteracion_Proyecto_idProyecto"),
145     @JoinColumn(name="numIteracionH",referencedColumnName="
            Iteracion_numIteracion")

147     },
        inverseJoinColumns={
149     @JoinColumn(name="numTarea",referencedColumnName="numTarea"),
        @JoinColumn(name="numProyecto",referencedColumnName="
            Iteracion_Proyecto_idProyecto"),
151     @JoinColumn(name="numIteracion",referencedColumnName="
            Iteracion_numIteracion")
    }
153 )
    public List<Tarea> getTareaPadre() {
155   return tareaPadre;
    }
157 public void setTareaPadre(List<Tarea> padre) {
      this.tareaPadre = padre;
159 }
    public int getpOpen() {
161   return pOpen;
    }
163 public void setpOpen(int pOpen) {
      this.pOpen = pOpen;
165 }
    public String getpCaption() {
167   return pCaption;
    }
169 public void setpCaption(String pCaption) {
      this.pCaption = pCaption;
171 }
    /**
173  * Los siguiente es un asociacion unidireccional. Es decir solo una
         isntacia
     * de Tarea sabe que tiene predecesores que son de tipo tarea, los
         predecesores
175  * no estan conscientes de ello.
     *
177  * Esta es un asociacion uno a muchos, la tabla de la relacion es '
         TareaPredecesor'
     * en esa tabla, la columna que tiene el identificador de esta
         instancia
179  * es 'idTarea', todas los predecesores estan en la columna '
         idPredecesor'
     * @return
```

```
181      */
         @OneToMany
183      @JoinTable(
             name="TareaPredecesor",
185          joinColumns={
                 @JoinColumn(name="numTarea",referencedColumnName="numTarea"),
187              @JoinColumn(name="numIteracion",referencedColumnName="
                     Iteracion_numIteracion"),
                 @JoinColumn(name="numProyecto",referencedColumnName="
                     Iteracion_Proyecto_idProyecto")
189          },
             inverseJoinColumns={
191              @JoinColumn(name="numTareaPredecesor",referencedColumnName="
                     numTarea"),
                 @JoinColumn(name="numIteracionP",referencedColumnName="
                     Iteracion_numIteracion"),
193              @JoinColumn(name="numProyectoP",referencedColumnName="
                     Iteracion_Proyecto_idProyecto")
             }
195      )
         public List<Tarea> getPredecesores() {
197        return predecesores;
         }
199      public void setPredecesores(List<Tarea> predecesores) {
           this.predecesores = predecesores;
201      }
         @OneToOne
203      @JoinColumn(name="Actividad_idActividad")
         public Actividad getActividad() {
205        return actividad;
         }
207      public void setActividad(Actividad actividad) {
           this.actividad = actividad;
209      }
         /**
211       * Un tarea esta asociada a varios usuarios
          * Debe ser un relacion bidireccional.
213       * (joinColumns={El identificador de este})
          * (inverseJoinColumns={El identificador del otro})
215       * @return
          */
217      @ManyToMany
         @JoinTable(name="TareaUsuario",
219          joinColumns={
               @JoinColumn(name="Tarea_numTarea",referencedColumnName="numTarea
                   "),
221            @JoinColumn(name="Tarea_Iteracion_numIteracion",
                   referencedColumnName="Iteracion_numIteracion"),
               @JoinColumn(name="Tarea_Iteracion_Proyecto_idProyecto",
                   referencedColumnName="Iteracion_Proyecto_idProyecto")
223        },
```

```
        inverseJoinColumns=@JoinColumn(name="Usuario_CURP",
            referencedColumnName="CURP")
225   )
    public Set<Usuario> getRecursos() {
227   return recursos;
    }
229   public void setRecursos(Set<Usuario> recursos) {
      this.recursos = recursos;
231   }
    /**
233  * Esta es complemento de la asociacion Padre-Hijo
     *
235  * @return
     */
237 @OneToMany
    @JoinTable(name="PadreTareaHijo",
239       joinColumns={
        @JoinColumn(name="numTarea",referencedColumnName="numTarea"),
241     @JoinColumn(name="numProyecto",referencedColumnName="
            Iteracion_Proyecto_idProyecto"),
        @JoinColumn(name="numIteracion",referencedColumnName="
            Iteracion_numIteracion")
243     },
        inverseJoinColumns={
245     @JoinColumn(name="numTareaHijo",referencedColumnName="numTarea"),
        @JoinColumn(name="numProyectoH",referencedColumnName="
            Iteracion_Proyecto_idProyecto"),
247     @JoinColumn(name="numIteracionH",referencedColumnName="
            Iteracion_numIteracion")
    }
249 )
    public List<Tarea> getSubtareas() {
251   return subtareas;
    }
253   public void setSubtareas(List<Tarea> subtareas) {
      this.subtareas = subtareas;
255   }

257   public boolean equals(Object other){
        if (this == other) return true;
259
        if ( !(other instanceof Tarea) ) return false;
261
        final Tarea t = (Tarea) other;
263
        if( t.getIdTarea().equals(this.getIdTarea()))
265       return true;
        else return false;
267   }
}
```

src/com/gestion/modelo/proyecto/Tarea.java

```java
package com.gestion.modelo.proyecto;

import java.io.Serializable;

import javax.persistence.Entity;
import javax.persistence.Id;


@Entity
public class Disciplina implements Serializable {
  private Integer idDisciplina;
  private String nombre;
  private String descripcion;

  public Disciplina(){

  }
  @Id
  public Integer getIdDisciplina() {
    return idDisciplina;
  }

  public void setIdDisciplina(Integer idDisciplina) {
    this.idDisciplina = idDisciplina;
  }

  public String getNombre() {
    return nombre;
  }

  public void setNombre(String nombre) {
    this.nombre = nombre;
  }

  public String getDescripcion() {
    return descripcion;
  }

  public void setDescripcion(String descripcion) {
    this.descripcion = descripcion;
  }
}
```

src/com/gestion/modelo/proyecto/Disciplina.java

```java
package com.gestion.modelo.proyecto;

import java.io.Serializable;
import java.sql.Date;
import java.util.ArrayList;
import java.util.HashSet;
import java.util.List;
```

```
8  import java.util.Set;

10 import javax.persistence.*;

12 import com.gestion.modelo.usuario.Usuario;

14 /*
    * Esta clase contiene la informacion relacionada con un proyecto.
16  */
   @Entity
18 public class Proyecto implements Serializable{
     private long idProyecto;
20   private String nombre;
     private String descripcion;
22   private String cliente;
     private Date fechaInicio;
24   private Date fechaFin;
     private Set<Usuario> listaUsuarios = new HashSet<Usuario>();
26   private Usuario jefe;
     private List<Iteracion> iteraciones = new ArrayList<Iteracion>();
28
     /*
30    * Constructor sin argumentos para Hibernate
      */
32   public Proyecto(){

34   }
     /*
36    * con la anotacion @Id indicamos que este atributo
      * es la clave primaria de la entidad Proyecto
38    */
     @Id
40   public long getIdProyecto() {
       return idProyecto;
42   }
     public void setIdProyecto(long idProyecto) {
44     this.idProyecto = idProyecto;
     }
46   public String getNombre() {
       return nombre;
48   }
     public void setNombre(String nombre) {
50     this.nombre = nombre;
     }
52   public String getDescripcion() {
       return descripcion;
54   }
     public void setDescripcion(String descripcion) {
56     this.descripcion = descripcion;
     }
58   public String getCliente() {
       return cliente;
```

```
60    }
      public void setCliente(String cliente) {
62      this.cliente = cliente;
      }
64
      /**
66     * Hay una asociacion Muchos a Muchos entre Proyecto y Usuario
       * @return
68     */
      @ManyToMany(targetEntity=Usuario.class)
70    @JoinTable(name="ParticipanteProyecto",joinColumns=@JoinColumn(name="
          Proyecto_idProyecto"),
           inverseJoinColumns=@JoinColumn(name="Usuario_CURP"))
72    public Set<Usuario> getListaUsuarios() {
        return listaUsuarios;
74    }
      public void setListaUsuarios(Set<Usuario> listaPersonal) {
76      this.listaUsuarios = listaPersonal;
      }
78
      //@Temporal(TemporalType.DATE)
80    public Date getFechaInicio() {
        return fechaInicio;
82    }
      public void setFechaInicio(Date fechaInicio) {
84      this.fechaInicio = fechaInicio;
      }
86    //@Temporal(TemporalType.DATE)
      public Date getFechaFin() {
88      return fechaFin;
      }
90    public void setFechaFin(Date fechaFin) {
        this.fechaFin = fechaFin;
92    }

94    /**
       * Un proyecto tiene solo un jefe de proyecto.
96     * con el codigo de abajo se ha establecido al relacion uno  a uno
       * Recupera tambien al Jefe
98     */
      @OneToOne
100   @JoinColumn(name="Usuario_CURP")
      public Usuario getJefe() {
102     return jefe;
      }
104   public void setJefe(Usuario jefe) {
        this.jefe = jefe;
106   }

108   /**
       * Un Proyecto esta asociado a varias Iteraciones
110    */
```

```
  @OneToMany(mappedBy="proyecto")
112 //@Column(name="id")
  public List<Iteracion> getIteraciones() {
114   return iteraciones;
  }
116
  public void setIteraciones(List<Iteracion> iteraciones) {
118   this.iteraciones = iteraciones;
  }
120
  /**
122  * Agrega un nuevo personal a lista de personal del proyecto.
   * @param nuevo Una instancia de Personal que se agregara al proyecto
       actual.
124  */
  public void agregarPersonal(Usuario nuevo){
126   this.listaUsuarios.add(nuevo);
  }
128
  public void eliminarPersonal(Usuario p){
130   if( this.listaUsuarios.contains(p)){
        System.out.println("Se eliminara: "+p.getIdUsuario());
132   }
  }
134 @Override
  public int hashCode() {
136   final int prime = 31;
    int result = 1;
138   result = prime * result + ((cliente == null) ? 0 : cliente.hashCode
        ());
    result = prime * result + (int) (idProyecto ^ (idProyecto >>> 32));
140   result = prime * result + ((nombre == null) ? 0 : nombre.hashCode())
        ;
    return result;
142 }
  @Override
144 public boolean equals(Object obj) {
    if (this == obj)
146   return true;
    if (obj == null)
148   return false;
    if (getClass() != obj.getClass())
150   return false;
    Proyecto other = (Proyecto) obj;
152 if (cliente == null) {
      if (other.cliente != null)
154     return false;
    } else if (!cliente.equals(other.cliente))
156   return false;
    if (idProyecto != other.idProyecto)
158   return false;
    if (nombre == null) {
```

```
160       if (other.nombre != null)
            return false;
162     } else if (!nombre.equals(other.nombre))
          return false;
164     return true;
    }
166


168


170 }
```

src/com/gestion/modelo/proyecto/Proyecto.java

```
  package com.gestion.modelo.proyecto;
2
  import java.io.Serializable;
4
  import javax.persistence.Column;
6 import javax.persistence.Embeddable;
  import javax.persistence.Transient;
8
  @Embeddable
10 public class IdTarea implements Serializable{
    private long numTarea;
12  private long numIteracion;
    private long idProyecto;
14  private String idCadena;

16  public IdTarea(){

18  }

20  public IdTarea(long idProyecto, long numIteracion, long numTarea){
      this.idProyecto = idProyecto;
22    this.numIteracion = numIteracion;
      this.numTarea = numTarea;
24  }

26  public long getNumTarea() {
      return numTarea;
28  }
    public void setNumTarea(long numTarea) {
30    this.numTarea = numTarea;
    }
32  @Column(name="Iteracion_numIteracion")
    public long getNumIteracion() {
34    return numIteracion;
    }
36  public void setNumIteracion(long numIteracion) {
      this.numIteracion = numIteracion;
38  }
    @Column(name="Iteracion_Proyecto_idProyecto")
```

```
40    public long getIdProyecto() {
        return idProyecto;
42    }
      public void setIdProyecto(long idProyecto) {
44      this.idProyecto = idProyecto;
      }
46    public String toString(){
        return idProyecto+""+numIteracion+""+numTarea;
48    }
      @Transient
50    public String getIdCadena() {
        return idProyecto+" "+numIteracion+" "+numTarea;
52    }
      public void setIdCadena(String id) {
54      this.idCadena = id;
      }
56
      public boolean equals(Object other){
58          if (this == other) return true;

60          if ( !(other instanceof IdTarea) ) return false;

62
            final IdTarea idO = (IdTarea) other;
64
            if( idO.getIdProyecto() == this.getIdProyecto() &&
66              idO.getNumIteracion() == this.getNumIteracion() &&
                idO.getNumTarea() == this.getNumTarea())
68            return true;
            else
70            return false;
      }
72  }
```

src/com/gestion/modelo/proyecto/IdTarea.java

```
    package com.gestion.modelo.proyecto;
2
    import java.io.Serializable;
4   import java.util.HashSet;
    import java.util.Set;
6
    import javax.persistence.Entity;
8   import javax.persistence.Id;
    import javax.persistence.JoinColumn;
10  import javax.persistence.JoinTable;
    import javax.persistence.ManyToMany;
12  import javax.persistence.OneToOne;

14  import com.gestion.modelo.rol.Rol;

16  @Entity
    public class Actividad implements Serializable{
```

```
18    private long idActividad;
      private Rol rol;
20    private Disciplina disciplina;
      private String nombre;
22    private String descripcion;
      private String urlDescripcion;
24    private Set<Artefacto> requeridos = new HashSet<Artefacto>();
      private Set<Artefacto> producidos = new HashSet<Artefacto>();
26    //agrgados el 2 de julio
      private String informacionGeneral;
28
      public Actividad(){
30
      }
32    @Id
      public long getIdActividad() {
34      return idActividad;
      }
36
      public void setIdActividad(long idActividad) {
38      this.idActividad = idActividad;
      }
40    @OneToOne
      @JoinColumn(name="Rol_nombreRol")
42    public Rol getRol() {
        return rol;
44    }

46    public void setRol(Rol rol) {
        this.rol = rol;
48    }
      @OneToOne
50    @JoinColumn(name="Disciplina_idDisciplina")
      public Disciplina getDisciplina() {
52      return disciplina;
      }
54
      public void setDisciplina(Disciplina disciplina) {
56      this.disciplina = disciplina;
      }
58
      public String getNombre() {
60      return nombre;
      }
62
      public void setNombre(String nombre) {
64      this.nombre = nombre;
      }
66
      public String getDescripcion() {
68      return descripcion;
      }
```

```
70
    public void setDescripcion(String descripcion) {
72    this.descripcion = descripcion;
    }
74
    public String getUrlDescripcion() {
76    return urlDescripcion;
    }
78
    public void setUrlDescripcion(String urlDescripcion) {
80    this.urlDescripcion = urlDescripcion;
    }
82  @ManyToMany
    @JoinTable(name="ArtefactoRequeridoActividad",
84      joinColumns=@JoinColumn(name="Actividad_idActividad",
            referencedColumnName="idActividad"),
        inverseJoinColumns=@JoinColumn(name="Artefacto_nombre",
            referencedColumnName="nombre")
86  )
    public Set<Artefacto> getRequeridos() {
88    return requeridos;
    }
90  public void setRequeridos(Set<Artefacto> requeridos) {
    this.requeridos = requeridos;
92  }
    @ManyToMany
94  @JoinTable(name="ArtefactoProducidoActividad",
      joinColumns=@JoinColumn(name="Actividad_idActividad"),
96    inverseJoinColumns=@JoinColumn(name="Artefacto_nombre")
    )
98  public Set<Artefacto> getProducidos() {
    return producidos;
100 }
    public void setProducidos(Set<Artefacto> producidos) {
102   this.producidos = producidos;
    }
104 public String getInformacionGeneral() {
    return informacionGeneral;
106 }
    public void setInformacionGeneral(String informacionGeneral) {
108   this.informacionGeneral = informacionGeneral;
    }
110
}
```

src/com/gestion/modelo/proyecto/Actividad.java

```
1 package com.gestion.modelo.rol;

3 import java.io.Serializable;

5 import javax.persistence.Column;
  import javax.persistence.Entity;
```

```
7 import javax.persistence.Id;
  @Entity
9 public class Rol implements Serializable{
    private String nombreRol;
11  private String descripcion;
    private String urlDescripcion;
13  private String categoria;

15  //agregado el 1 de julio 00:29 am
    private String habilidades;
17  private String criterioAsignacion;

19  public Rol(){

21  }
    @Id
23  public String getNombreRol() {
      return nombreRol;
25  }
    public void setNombreRol(String nombreRol) {
27    this.nombreRol = nombreRol;
    }
29
    public String getDescripcion() {
31    return descripcion;
    }
33  public void setDescripcion(String descripcion) {
      this.descripcion = descripcion;
35  }
    public String getUrlDescripcion() {
37    return urlDescripcion;
    }
39  public void setUrlDescripcion(String urlDescripcion) {
      this.urlDescripcion = urlDescripcion;
41  }
    public String getCategoria() {
43    return categoria;
    }
45  public void setCategoria(String categoria) {
      this.categoria = categoria;
47  }
    public String getHabilidades() {
49    return habilidades;
    }
51  public void setHabilidades(String habilidades) {
      this.habilidades = habilidades;
53  }
    public String getCriterioAsignacion() {
55    return criterioAsignacion;
    }
57  public void setCriterioAsignacion(String criterioAsignacion) {
      this.criterioAsignacion = criterioAsignacion;
```

```
59    }
      @Override
61    public int hashCode() {
        final int prime = 31;
63      int result = 1;
        result = prime * result
65          + ((categoria == null) ? 0 : categoria.hashCode());
        result = prime * result
67          + ((nombreRol == null) ? 0 : nombreRol.hashCode());
        result = prime * result
69          + ((urlDescripcion == null) ? 0 : urlDescripcion.hashCode());
        return result;
71    }
      @Override
73    public boolean equals(Object obj) {
        if (this == obj)
75        return true;
        if (obj == null)
77        return false;
        if (getClass() != obj.getClass())
79        return false;
        Rol other = (Rol) obj;
81      if (categoria == null) {
          if (other.categoria != null)
83          return false;
        } else if (!categoria.equals(other.categoria))
85        return false;
        if (nombreRol == null) {
87        if (other.nombreRol != null)
            return false;
89      } else if (!nombreRol.equals(other.nombreRol))
          return false;
91      if (urlDescripcion == null) {
          if (other.urlDescripcion != null)
93          return false;
        } else if (!urlDescripcion.equals(other.urlDescripcion))
95        return false;
        return true;
97    }
    }
```

src/com/gestion/modelo/rol/Rol.java

```
    package com.gestion.modelo.manejadorBd.proyecto;
2


4   import java.util.List;


6


    import org.hibernate.HibernateException;
8   import org.hibernate.Query;

10  import org.hibernate.classic.Session;
```

```
12 import com.gestion.modelo.proyecto.Proyecto;
   import com.gestion.modelo.usuario.Usuario;
14 import com.gestion.util.HibernateUtil;

16 public class ManejadorProyecto extends HibernateUtil{
     /**
18    * Consulta la tabla Proyectos y crea una lista de todas
      * los proyectos.
20    * @return List<Proyecto>
      */
22    public List<Proyecto> lista(){
        Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
24      session.beginTransaction();
        List<Proyecto> proyectos = null;
26      try {
          /*
28         * la consulta en hql "from Proyecto"
           * retorna todas las instancias de Proyecto
30         */
          proyectos = (List<Proyecto>)session.createQuery("from Proyecto").
              list();
32      } catch (HibernateException e) {
          e.printStackTrace();
34        session.getTransaction().rollback();
        }
36      session.getTransaction().commit();
        return proyectos;
38    }
      /**
40    * Obtiene la lista de proyectos en los cuales el
      * usuario con idUsuario es jefe de proyecto.
42    * @param idUsuario
      * @return
44    */
      public List<Proyecto> listaProyectos(String idUsuario){
46      Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
        session.beginTransaction();
48      List<Proyecto> proyectos = null;
        Query query;
50      try {
          System.out.println("IDUSUARIO: "+idUsuario);
52        query = session.createSQLQuery("select * from Proyecto where
              Usuario_CURP=?").addEntity(Proyecto.class);
          proyectos = (List<Proyecto>) query.setString(0, idUsuario+"").list
              ();
54      } catch (HibernateException e) {
          e.printStackTrace();
56        session.getTransaction().rollback();
        }
```

```
58      session.getTransaction().commit();
        return proyectos;
60    }
    /**
62   * Agrega un nuevo proyecto a la base de datos.
     * @param nuevo, una instancia de Proyecto
64   * @return Proyecto
     */
66  public Proyecto registrarProyecto(Proyecto nuevo){
      Session session = HibernateUtil.getSessionFactory().
          getCurrentSession();
68    session.beginTransaction();
      session.save(nuevo);
70    session.getTransaction().commit();
      return nuevo;
72  }

74  /**
     * Obtiene una instancia de Proyecto con el identificador dado.
76   * @param idProyecto Identificador del proyecto.
     * @return Una instancia del Proyecto con el identificador dado.
78   */
    public Proyecto datosProyecto(long idProyecto){
80    Session session = HibernateUtil.getSessionFactory().
          getCurrentSession();
      session.beginTransaction();
82    Proyecto proyecto = (Proyecto)session.get(Proyecto.class, idProyecto
          );
      session.getTransaction().commit();
84    return proyecto;
    }
86  /**
     * Elimina el proyecto con el identificador recibido.
88   * @param idProyecto El identificador del proyecto a eliminar.
     * @return Una instancia del Proyecto eliminado.
90   */
    public Proyecto eliminarProyecto(long idProyecto){
92    Session session = HibernateUtil.getSessionFactory().
          getCurrentSession();
      session.beginTransaction();
94    Proyecto proyecto = (Proyecto) session.get(Proyecto.class,
          idProyecto);
      if(null != proyecto) {
96      session.delete(proyecto);
      }
98    session.getTransaction().commit();
      return proyecto;
100 }
    /**
102  * Guarda o actualiza un proyecto.
     * @param proyecto La instancia del proyecto a actualizar.
104  * @return El proyecto actualizado.
```

```
     */
106  public Proyecto actualizarProyecto(Proyecto proyecto){
       Session session= HibernateUtil.getSessionFactory().getCurrentSession
           ();
108    session.beginTransaction();
       session.update(proyecto);
110    session.getTransaction().commit();
       return proyecto;
112  }
     /**
114   * Obtiene las iteraciones del proyecto que tiene el identificador
          idProyecto
      * si es que existe
116   * @param idProyecto
      * @return
118   */
     public Proyecto iteraciones(long idProyecto){
120    Session session = HibernateUtil.getSessionFactory().
           getCurrentSession();
       session.beginTransaction();
122    Proyecto proyecto = null;
       try{
124
       }catch(Exception e){
126      proyecto = (Proyecto)session.get(Proyecto.class, idProyecto);
         proyecto.getIteraciones().size();
128    }
       session.getTransaction().commit();
130    return proyecto;
     }
132  /**
      * Carga las iteraciones asociadas con proyecto.
134   * @param proyecto
      * @return
136   */
     public Proyecto cargarIteraciones(Proyecto proyecto){
138    Session session = HibernateUtil.getSessionFactory().
           getCurrentSession();
       session.beginTransaction();
140    try{
         proyecto = (Proyecto)session.load(Proyecto.class, proyecto.
             getIdProyecto());
142      proyecto.getIteraciones().size();
       }catch(Exception e){
144      e.printStackTrace();
       }
146    session.getTransaction().commit();
       return proyecto;
148  }
     /**
150   * Carga los usuarios asociados con proyecto.
      * @param proyecto
```

```
152     * @return
        */
154   public Proyecto cargarUsuarios(Proyecto proyecto){
        Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
156     session.beginTransaction();
        try{
158       proyecto = (Proyecto)session.load(Proyecto.class, proyecto.
              getIdProyecto());
          proyecto.getListaUsuarios().size();
160       System.out.println("Lista: "+proyecto.getListaUsuarios().size());
        }catch(Exception e){
162       e.printStackTrace();
        }
164     session.getTransaction().commit();
        return proyecto;
166   }

168   public Proyecto cargarAsociaciones(Proyecto proyecto){
        Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
170     session.beginTransaction();
        try{
172       proyecto = (Proyecto)session.load(Proyecto.class, proyecto.
              getIdProyecto());
          proyecto.getIteraciones().size();
174       proyecto.getListaUsuarios().size();
        }catch(Exception e){
176       e.printStackTrace();
        }
178     session.getTransaction().commit();
        return proyecto;
180   }
}
```

src/com/gestion/modelo/manejadorBd/proyecto/ManejadorProyecto.java

```
1 package com.gestion.modelo.manejadorBd.rol;

3 import java.util.List;

5 import org.hibernate.classic.Session;


7
  import com.gestion.modelo.rol.Rol;
9 import com.gestion.util.HibernateUtil;

11 public class ManejadorRol extends HibernateUtil{
     /**
13    * Recupera los roles en una lista.
      * @return
15    */
     public List<Rol> lista(){
```

```
17      Session session = new HibernateUtil().getSessionFactory().
            getCurrentSession();
        session.beginTransaction();
19      List<Rol> roles = null;
        try{
21        roles = (List<Rol>)session.createQuery("from Rol").list();
        }catch(Exception e){
23        e.printStackTrace();
          session.getTransaction().rollback();
25      }
        session.getTransaction().commit();
27      return roles;
    }
29  /**
     * Agregar un rol
31   * @param rol
     * @return
33   */
    public Rol add(Rol rol){
35    Session session = HibernateUtil.getSessionFactory().
          getCurrentSession();
      session.beginTransaction();
37    session.save(rol);
      session.getTransaction().commit();
39    return rol;
    }
41  /**
     * Buscar un rol con el nombre dado.
43   * @param nombre_rol
     * @return
45   */
    public Rol find(String nombre_rol){
47    Session session = HibernateUtil.getSessionFactory().
          getCurrentSession();
      session.beginTransaction();
49    Rol rol = (Rol)session.get(Rol.class, nombre_rol);
      session.getTransaction().commit();
51    return rol;
    }
53  /**
     * Actualizar el rol especificado
55   * @param rol
     * @return
57   */
    public Rol actualizarRol(Rol rol){
59    Session session = HibernateUtil.getSessionFactory().
          getCurrentSession();
      session.beginTransaction();
61    session.update(rol);
      session.getTransaction().commit();
63    return rol;
    }
```

```
65    /**
       * Elimina de la base de datos el rol especificado
67     * @param rol
       * @return
69     */
      public Rol eliminarRol(Rol rol){
71      Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
        session.beginTransaction();
73      session.delete(rol);
        session.getTransaction().commit();
75      return rol;
      }
77  }
```

src/com/gestion/modelo/manejadorBd/rol/ManejadorRol.java

```
1   package com.gestion.modelo.manejadorBd.usuario;

3   import java.util.ArrayList;
    import java.util.Iterator;
5   import java.util.List;
    import java.util.Set;
7
    import org.apache.catalina.User;
9   import org.hibernate.HibernateException;
    import org.hibernate.Query;
11  import org.hibernate.SQLQuery;
    import org.hibernate.classic.Session;
13
    import com.gestion.modelo.rol.Rol;
15  import com.gestion.modelo.usuario.Accion;
    import com.gestion.modelo.usuario.IdPermiso;
17  import com.gestion.modelo.usuario.Permiso;
    import com.gestion.modelo.usuario.Usuario;
19  import com.gestion.util.HibernateUtil;
    import com.gestion.util.PermisoDTO;
21
    public class ManejadorUsuario extends HibernateUtil{
23    /**
       * Recuperar todas las instancias del personal
25     * existente
       */
27    public List<Usuario> lista(){
        Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
29      session.beginTransaction();
        List<Usuario> personal = null;
31      try {
          /*
33         * la consulta en hql "from Proyecto"
           * retorna todas las instancias de Proyecto
35         */
```

```
      personal = (List<Usuario>)session.createQuery("from Usuario").list
          ();

    } catch (HibernateException e) {
      e.printStackTrace();
      session.getTransaction().rollback();
    }
    session.getTransaction().commit();
    return personal;
  }
  /**
   * Recupera todas las instancias de Personal del proyecto con el
       identificador
   * dado.
   * @param idProyecto Identificador del proyecto.
   * @return La lista con las instancias Personal relacionadas con el
       proyecto
   * que tiene el identificador dado.
   */
  public List<Usuario> listaUsuarios(long idProyecto){
    Session session = HibernateUtil.getSessionFactory().
        getCurrentSession();
    session.beginTransaction();
    List<Usuario> Empleado = null;
    try {
      /*
       * la consulta en hql "from Proyecto"
       * retorna todas las instancias de Proyecto
       */
      Empleado = (List<Usuario>)session.createQuery("from Usuario").list
          ();
      //Empleado = (List<Empleado>)session.createCriteria(Empleado.class
          ).
          //add(Restrictions.eq(propertyName, value));

    } catch (HibernateException e) {
      e.printStackTrace();
      session.getTransaction().rollback();
    }
    session.getTransaction().commit();
    return Empleado;
  }

  /**
   * Guarda en la base de datos una instancia de Empleado.
   * @param nuevo La instancia a ser almacenada.
   * @return La instancia almacenada.
   */
  public Usuario add(Usuario nuevo){
    Session session = HibernateUtil.getSessionFactory().
        getCurrentSession();
    session.beginTransaction();
```

```
81      session.save(nuevo);
        session.getTransaction().commit();
83      return nuevo;
    }
85
    /**
87   * Busca en la base de datos un Empleado con el identificador recibido
         .
     * @param idEmpleado El identificador del Empleado a buscar.
89   * @return Una instancia de Empleado que tiene el identificador
         recibido.
     */
91  public Usuario buscar(String idEmpleado){
      Session session = HibernateUtil.getSessionFactory().
          getCurrentSession();
93    session.beginTransaction();
      Usuario usuario = null;
95    try{
        usuario = (Usuario)session.get(Usuario.class, idEmpleado);
97      if( usuario != null) System.out.println("Encontrado: "+usuario.
            getNombreCompleto());
      }catch(Exception e){
99      e.printStackTrace();
      }
101   session.getTransaction().commit();
      return usuario;
103 }
    /**
105  * Regresa un conjunto de instancias de Empleado que pueden ser
     * asignados al proyecto con el identificador dado.
107  * @return
     */
109 public List<Usuario> EmpleadoDisponible(long idProyecto){
      Session session = HibernateUtil.getSessionFactory().
          getCurrentSession();
111   session.beginTransaction();
      List<Usuario> EmpleadoDisponible = new ArrayList<Usuario>();
113   Query query;
      try{
115     query = session.createSQLQuery("select * from Usuario where CURP "
            +
            "<> all (select Usuario_CURP from ParticipanteProyecto " +
117         "where Proyecto_idProyecto=?)").addEntity(Usuario.class);
        EmpleadoDisponible = (List<Usuario>) query.setString(0, idProyecto
            +"").list();
119     System.out.println("Empleado disponible: "+EmpleadoDisponible.size
            () );
      }catch(Exception e){
121     e.printStackTrace();
      }
123   session.getTransaction().commit();
      return EmpleadoDisponible;
```

```
125   }

127   /**
       * Elimina de la base de datos el usuario identificado con id.
129    * @param id El identificador del usuario.
       * @return El usuario eliminado.
131    */
      public Usuario eliminarUsuario(String id) {
133     Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
        session.beginTransaction();
135     Usuario usuario = (Usuario) session.load(Usuario.class, id);
        if(null != usuario) {
137       session.delete(usuario);
        }
139     session.getTransaction().commit();
        return usuario;
141   }
      /**
143    * Actualiza en la bd una instancia de usuario.
       * @param usuario La instancia de usuario a ser actualizada.
145    * @return El usuario actualizado.
       */
147   public Usuario actualizarUsuario(Usuario usuario){
        Session session= HibernateUtil.getSessionFactory().getCurrentSession
            ();
149     session.beginTransaction();
        session.saveOrUpdate(usuario);
151     session.getTransaction().commit();
        return usuario;
153   }
      /**
155    * Carga los usuarios asociados con proyecto que tiene el rol XXX
       * @param proyecto
157    * @return
       */
159   public List<Usuario> usuariosConRol(long idProyecto, String rol){
        Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
161     session.beginTransaction();
        Query query;
163     List<Usuario> usuarios=null;
        try{
165       query = session.createSQLQuery("select CURP,nombre,ape_paterno,
              ape_materno,telefono,correo,login,pass_word from " +
              "(select * from Usuario inner join " +
167             "(select Usuario_CURP from ParticipanteProyecto where
                  Proyecto_idProyecto=?) as b " +
              "where Usuario.CURP = b.Usuario_CURP) as c " +
169           "inner join UsuarioRol as ur " +
              "where ur.Usuario_CURP=c.CURP and ur.Rol_nombreRol=?");
171       query.setString(0, idProyecto+"");
```

```
        query.setString(1, rol);
173     usuarios = (List<Usuario>)query.list();
      }catch(Exception e){
175     e.printStackTrace();
      }
177   session.getTransaction().commit();
      return usuarios;
179 }

181 /**
     * Carga los usuarios del sistema que pueden desempenar el rol XXX
183  * @param proyecto
     * @return
185  */
    public List<Usuario> usuariosSistemaRol(String rol){
187   Session session = HibernateUtil.getSessionFactory().
        getCurrentSession();
      session.beginTransaction();
189   Query query;
      List<Usuario> usuarios=null;
191   try{
        query = session.createSQLQuery("Select CURP,nombre,ape_paterno,
            ape_materno,telefono, correo,login,pass_word from Usuario
            inner join (select Usuario_CURP from UsuarioRol where
            Rol_nombreRol=?) as b where Usuario.CURP=b.Usuario_CURP;").
            addEntity(Usuario.class);
193     //query.addEntity(Usuario.class);
        query.setString(0, rol);
195     usuarios = (List<Usuario>)query.list();
      }catch(Exception e){
197     e.printStackTrace();
      }
199   session.getTransaction().commit();
      return usuarios;
201 }

203

    /**
205  * Consulta los roles del usuario especificado
     * @param user
207  * @return
     */
209 public Usuario cargarRoles(Usuario user){
      Session session= HibernateUtil.getSessionFactory().getCurrentSession
          ();
211   session.beginTransaction();
      try{
213     user = (Usuario)session.load(Usuario.class, user.getIdUsuario());
        user.getRoles().size();
215     System.out.println(user.getRoles().size());
      }catch(Exception e){
217     e.printStackTrace();
```

```
    }
219     session.getTransaction().commit();
    return user;
221   }
    /**
223    * Consulta en la bd al usuario con el correo dado
     */
225   public Usuario usuario(String correo){
    Session session = HibernateUtil.getSessionFactory().
        getCurrentSession();
227     session.beginTransaction();
    SQLQuery query;
229     Usuario user = null;
    try{
231       List<Usuario> usuarios;
      query = session.createSQLQuery("select * from Usuario where correo
          =?");
233       query.addEntity(Usuario.class);
      query.setString(0, correo);
235       usuarios = (List<Usuario>)query.list();
      if(!usuarios.isEmpty())
237         user = usuarios.get(0);
      else return null;
239     }catch(Exception e){
      e.printStackTrace();
241     }
    session.getTransaction().commit();
243     return user;
    }
245   /**
     * Carga los proyectos del usuario que recibe como argumento.
247    * @param usr
     * @return
249    */
    public Usuario cargarProyectos(Usuario usr){
251     Session session = HibernateUtil.getSessionFactory().
        getCurrentSession();
    session.beginTransaction();
253     try{
      usr = (Usuario)session.get(Usuario.class, usr.getIdUsuario());
255       usr.getProyectos().size();
      System.out.println(" Proyectos del usuario: " + usr.getIdUsuario()
          +" "+usr.getProyectos().size());
257     }catch(Exception e){
      e.printStackTrace();
259       usr = null;
    }
261     session.getTransaction().commit();
    return usr;
263   }
    /**
265    * Carga las tareas del usuario
```

```
     * @param usr
267  * @return
     */
269  public Usuario cargarTareasUsuario(Usuario usr){
       Session session = HibernateUtil.getSessionFactory().
           getCurrentSession();
271    session.beginTransaction();
       try{
273      usr = (Usuario)session.get(Usuario.class, usr.getIdUsuario());
         usr.getTareas().size();
275    }catch(Exception e){
         e.printStackTrace();
277      usr = null;
         session.getTransaction().rollback();
279    }
       session.getTransaction().commit();
281    return usr;
     }
283
     public boolean permitido(Set<Rol> roles, String accion ){
285    StringBuffer rls = new StringBuffer();
       StringBuffer sql = new StringBuffer();
287
       Iterator<Rol> it = roles.iterator();
289    boolean siguiente = it.hasNext();
       while(siguiente){
291      Rol r = it.next();
         rls.append("'"+r.getNombreRol()+"'");
293      siguiente = it.hasNext();
         if(siguiente) rls.append(",");
295    }

297    if(rls.toString().trim().length()==0) rls.append("''");

299    sql.append(" select p.idAccion, p.rol, p.permitido, a.nombre from
           Permiso as p left outer join Accion as a  on p.idAccion = a.id")
           ;
       sql.append(" where p.rol in ("+rls.toString()+") and a.nombre ='"+
           accion+"' and p.permitido=1");
301
       Session session = HibernateUtil.getSessionFactory().
           getCurrentSession();
303    session.beginTransaction();
       SQLQuery query;
305    boolean res = false;
       try{
307      System.out.println("Consulta: "+sql.toString());
         query = session.createSQLQuery(sql.toString());
309      if(query.list().size()>0) res = true;
         else res = false;
311    }catch(Exception e){
         e.printStackTrace();
```

```
313     }
        session.getTransaction().commit();
315     return res;

317   }

319   public List<Accion> listaAcciones (){
        Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
321     session.beginTransaction();
        List<Accion> acciones = null;
323     try {
          acciones = (List<Accion>)session.createQuery("from Accion order by
              categoria").list();
325
        } catch (HibernateException e) {
327       e.printStackTrace();
          session.getTransaction().rollback();
329     }
        session.getTransaction().commit();
331     return acciones;
      }
333
      public List<Permiso> permisos(){
335     Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
        session.beginTransaction();
337     List<Permiso> permisos = null;
        try {
339       permisos = (List<Permiso>)session.createQuery("from Permiso").list
              ();

341     } catch (HibernateException e) {
          e.printStackTrace();
343       session.getTransaction().rollback();
        }
345     session.getTransaction().commit();
        return permisos;
347   }

349   public List<Permiso> permisosXRol(String rol){
        Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
351     session.beginTransaction();
        List<Permiso> permisos = null;
353     try {
          permisos = (List<Permiso>)session.createQuery("from Permiso where
              rol.nombreRol ='"+rol+"'").list();
355
        } catch (HibernateException e) {
357       e.printStackTrace();
          session.getTransaction().rollback();
```

```
359       }
          session.getTransaction().commit();
361       return permisos;
      }
363
      public void guardaObjeto(Object obj){
365       Session session = HibernateUtil.getSessionFactory().
              getCurrentSession();
          session.beginTransaction();
367       try {
            session.save(obj);
369       } catch (HibernateException e) {
            e.printStackTrace();
371         session.getTransaction().rollback();
          }
373       session.getTransaction().commit();
      }
375
      public void deleteObject(Object obj){
377       Session session = HibernateUtil.getSessionFactory().
              getCurrentSession();
          session.beginTransaction();
379       try {
            session.delete(obj);
381       } catch (HibernateException e) {
            e.printStackTrace();
383         session.getTransaction().rollback();
          }
385       session.getTransaction().commit();
      }
387
      public Accion getAccion(Integer id){
389       Session session = HibernateUtil.getSessionFactory().
              getCurrentSession();
          session.beginTransaction();
391       Accion a = null;
          try {
393         a = (Accion)session.get(Accion.class,id);
          } catch (HibernateException e) {
395         e.printStackTrace();
            session.getTransaction().rollback();
397       }
          session.getTransaction().commit();
399       return a;
      }
401
      public void actualizaObjeto(Object obj){
403       Session session = HibernateUtil.getSessionFactory().
              getCurrentSession();
          session.beginTransaction();
405       try {
            session.update(obj);
```

```
407     } catch (HibernateException e) {
          e.printStackTrace();
409       session.getTransaction().rollback();
        }
411     session.getTransaction().commit();
    }
413
    public List<PermisoDTO> permisosDTOXRol(String rol){
415   if(rol==null || rol.trim().length()==0) return new ArrayList<
          PermisoDTO>();
      StringBuffer sql = new StringBuffer();
417   List<PermisoDTO> permisosDTO = new ArrayList<PermisoDTO>();
      sql.append("select A.id, A.nombre, A.descripcion, A.categoria, P.rol
          , P.permitido from Accion as A left outer join Permiso as P on A
          .id = P.idAccion and P.rol = '"+rol+"'");
419


421   Session session = HibernateUtil.getSessionFactory().
          getCurrentSession();
      session.beginTransaction();
423   SQLQuery query;
      try{
425     query = session.createSQLQuery(sql.toString());
        List obj = query.list();
427     String rr = null;
        for(Object o: obj ){
429       Object[] ar = (Object[])o;
          rr = (String)ar[4];
431       if(rr == null) rr = rol;
          permisosDTO.add(new PermisoDTO(ar[0],ar[1],ar[2],ar[3],rr,ar[5])
              );
433     }

435   }catch(Exception e){
        e.printStackTrace();
437   }
      session.getTransaction().commit();
439   return permisosDTO;
    }
441
    public void cambiarPermiso(Integer idAccion, String rol){
443   Session session = HibernateUtil.getSessionFactory().
          getCurrentSession();
      session.beginTransaction();
445   try {
        IdPermiso idP = new IdPermiso();
447     idP.setIdAccion(Integer.valueOf(idAccion));
        idP.setRol(rol);
449     Permiso p = (Permiso) session.get(Permiso.class,idP);
        if(p==null){
451       Accion a = (Accion) session.get(Accion.class, idAccion);
          Rol r = (Rol)session.get(Rol.class, rol);
```

```
453             Permiso nuevo = new Permiso();
                nuevo.setAccion(a);
455             nuevo.setRol(r);
                nuevo.setPermitido(true);
457             nuevo.setId(idP);
                session.saveOrUpdate(nuevo);
459             System.out.println("Permiso creado en la bd");
            }else{
461             Boolean a = p.getPermitido();
                Boolean n = p.getPermitido()==null?false:!p.getPermitido().
                    booleanValue();
463             System.out.println("Permiso actual: "+p.getPermitido());
                System.out.println("Nuevo valor del permiso: "+n);
465             System.out.println("IdAccion:"+p.getId().getIdAccion());
                System.out.println("Rol: "+p.getId().getRol());
467             System.out.println("Permitido:"+p.getPermitido());
                p.setPermitido(n);
469             session.update(p);
                System.out.println("Permiso actualizado a: "+p.getPermitido());
471         }
        } catch (HibernateException e) {
473         e.printStackTrace();
            session.getTransaction().rollback();
475     }
        session.getTransaction().commit();
477     return;
    }
479 }
```

src/com/gestion/modelo/manejadorBd/usuario/ManejadorUsuario.java

```
1  package com.gestion.modelo.manejadorBd.fase;

3  import java.util.List;

5  import org.hibernate.Query;
   import org.hibernate.Session;
7
   import com.gestion.modelo.proyecto.Actividad;
9  import com.gestion.util.HibernateUtil;

11 public class ManejadorActividad {
     /**
13    * Recupera la lista de actividades disponibles.
      * @return
15    */
     public List<Actividad> lista(){
17     Session session = HibernateUtil.getSessionFactory().
           getCurrentSession();
       session.beginTransaction();
19     List<Actividad> actividades = null;
       try{
```

```
21        actividades = (List<Actividad>)session.createQuery("FROM Actividad
              ").list();
      }catch(Exception e){
23        e.printStackTrace();
          session.getTransaction().rollback();
25    }
      session.getTransaction().commit();
27    return actividades;
  }
29  /**
   * Recupera la lista de actividades que pertenecen a la actividad
31 * @return
   */
33  public List<Actividad> lista(long idDisciplina){
      Session session = HibernateUtil.getSessionFactory().
          getCurrentSession();
35    session.beginTransaction();
      List<Actividad> actividades = null;
37    Query query =null;
      try{
39      query = session.createSQLQuery("select * from Actividad where
            Disciplina_idDisciplina=?").addEntity(Actividad.class);
        query.setString(0, idDisciplina+"");
41      actividades = query.list();
      }catch(Exception e){
43      e.printStackTrace();
        session.getTransaction().rollback();
45    }
      session.getTransaction().commit();
47    return actividades;
  }
49  /**
   * Agrega una actividad en el sistema.
51 * @param actividad
   * @return
53 */
  public Actividad agregarActividad(Actividad actividad){
55    Session session = HibernateUtil.getSessionFactory().
          getCurrentSession();
      session.beginTransaction();
57    try{
        session.save(actividad);
59    }catch(Exception e){
        e.printStackTrace();
61      session.getTransaction().rollback();
        return null;
63    }
      session.getTransaction().commit();
65    return actividad;
  }
67  /**
   * Actualiza la actividad en la base de datos.
```

```
69      * @param actividad
        * @return
71      */
    public Actividad actualizarActividad(Actividad actividad){
73      Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
        session.beginTransaction();
75      try{
          session.update(actividad);
77      }catch(Exception e){
          e.printStackTrace();
79        session.getTransaction().rollback();
          return null;
81      }
        session.getTransaction().commit();
83      return actividad;
    }
85  /**
     * Elimina la actividad de la base de datos.
87   * @param actividad
     * @return
89   */
    public Actividad eliminarActividad(Actividad actividad){
91      Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
        session.beginTransaction();
93      try{
          session.delete(actividad);
95      }catch(Exception e){
          e.printStackTrace();
97        session.getTransaction().rollback();
        }
99      session.getTransaction().commit();
        return actividad;
101 }
    /**
103  * Carga los artefactos requeridos de la actividad.
     * @param actividad
105  * @return
     */
107 public Actividad cargarArtefactosRequeridos(Actividad actividad){
        Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
109     session.beginTransaction();
        try{
111       actividad = (Actividad)session.load(Actividad.class, actividad.
              getIdActividad());
          actividad.getRequeridos().size();
113     }catch(Exception e){
          e.printStackTrace();
115       session.getTransaction().rollback();
        }
```

```
117     session.getTransaction().commit();
        return actividad;
119   }
      /**
121    * Carga los artefactos producidos en la actividad.
       * @param actividad
123    * @return
       */
125   public Actividad cargarArtefactosProducidos(Actividad actividad){
        Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
127     session.beginTransaction();
        try{
129       actividad = (Actividad)session.load(Actividad.class, actividad.
              getIdActividad());
          actividad.getProducidos().size();
131     }catch(Exception e){
          e.printStackTrace();
133       session.getTransaction().rollback();
        }
135     session.getTransaction().commit();
        return actividad;
137   }
      /**
139    * Busca la actividad con el id especificado
       * @param idActividad
141    * @return
       */
143   public Actividad buscarActividad(long idActividad){
        Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
145     session.beginTransaction();
        Actividad actividad=null;
147     try{
          actividad= (Actividad)session.get(Actividad.class, idActividad);
149     }catch(Exception e){
          e.printStackTrace();
151       session.getTransaction().rollback();
        }
153     session.getTransaction().commit();
        return actividad;
155   }

157   public Actividad cargarArtefactos(Actividad actividad){
        Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
159     session.beginTransaction();
        try{
161       actividad = (Actividad)session.load(Actividad.class, actividad.
              getIdActividad());
          actividad.getProducidos().size();
163       actividad.getRequeridos().size();
```

```
    }catch(Exception e){
      e.printStackTrace();
      session.getTransaction().rollback();
    }
    session.getTransaction().commit();
    return actividad;
  }
}
```

src/com/gestion/modelo/manejadorBd/fase/ManejadorActividad.java

```
package com.gestion.modelo.manejadorBd.fase;

import java.util.List;

import org.hibernate.HibernateException;
import org.hibernate.Query;
import org.hibernate.classic.Session;

import com.gestion.modelo.proyecto.Fase;
import com.gestion.modelo.proyecto.Iteracion;
import com.gestion.util.HibernateUtil;

public class ManejadorFaseIteracion extends HibernateUtil{
  public List<Fase> lista(){
    Session session = HibernateUtil.getSessionFactory().
        getCurrentSession();
    session.beginTransaction();
    List<Fase> fases = null;
    try {
      fases = (List<Fase>)session.createQuery("from Fase").list();
    } catch (HibernateException e) {
      e.printStackTrace();
      session.getTransaction().rollback();
    }
    session.getTransaction().commit();
    return fases;
  }
  public Fase fase(int id){
    Session session = HibernateUtil.getSessionFactory().
        getCurrentSession();
    session.beginTransaction();
    Fase f=null;
    try{
      f = (Fase)session.get(Fase.class, id);
    }catch(HibernateException e){
      e.printStackTrace();
      session.getTransaction().rollback();
    }
    session.getTransaction().commit();
    return f;
  }
```

```
41   public List<Iteracion> iteracionesProyecto(long idProyecto, String
        fase){
     Session session = HibernateUtil.getSessionFactory().
        getCurrentSession();
43   session.beginTransaction();
     List<Iteracion> iteraciones= null;
45   Query query;
     try {
47     query = session.createSQLQuery("SELECT * FROM Iteracion WHERE
          Fase_Proyecto_idProyecto=?" +
          " AND Fase_nombre=?").addEntity(Iteracion.class);
49     query.setString(1, fase);
       iteraciones = (List<Iteracion>) query.setString(0, idProyecto+"").
          list();
51   } catch (HibernateException e) {
       e.printStackTrace();
53     session.getTransaction().rollback();
     }
55   session.getTransaction().commit();
     return iteraciones;
57 }
}
```

src/com/gestion/modelo/manejadorBd/fase/ManejadorFaseIteracion.java

```
   package com.gestion.modelo.manejadorBd.fase;
2
   import java.util.ArrayList;
4  import java.util.Iterator;
   import java.util.List;
6  import java.util.Set;

8  import org.hibernate.HibernateException;
   import org.hibernate.Query;
10 import org.hibernate.SQLQuery;
   import org.hibernate.classic.Session;
12 import org.hibernate.mapping.Array;

14 import com.gestion.modelo.proyecto.IdTarea;
   import com.gestion.modelo.proyecto.Iteracion;
16 import com.gestion.modelo.proyecto.Proyecto;
   import com.gestion.modelo.proyecto.Tarea;
18 import com.gestion.util.HibernateUtil;

20 public class ManejadorTarea {
     /**
22    * Consulta la tabla Tareas y crea una lista de
      * las tareas.
24    * @return List<Tarea>
      */
26   public List<Tarea> lista(){
       Session session = HibernateUtil.getSessionFactory().
          getCurrentSession();
```

```
28      session.beginTransaction();
        List<Tarea> tareas = null;
30      try {
          tareas = (List<Tarea>)session.createQuery("from Tarea").list();
32      } catch (HibernateException e) {
          e.printStackTrace();
34        session.getTransaction().rollback();
        }
36      session.getTransaction().commit();
        return tareas;
38    }

40    /**
       * Recupera la tarea que tiene el identificador idTarea
42     * @param idTarea Identificador de la tarea.
       * @return Un objeto Tarea.
44     */
      public Tarea datosTarea(IdTarea idTarea){
46      Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
        session.beginTransaction();
48      Tarea tarea = (Tarea)session.get(Tarea.class, idTarea);
        session.getTransaction().commit();
50      return tarea;
      }
52    /**
       * Agrega una nueva tarea
54     * @param tarea
       * @return
56     */
      public Tarea agregarTarea(Tarea tarea){
58      Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
        session.beginTransaction();
60      try{
          session.save(tarea);
62      }catch(Exception e){
          e.printStackTrace();
64      }
        session.getTransaction().commit();
66      return tarea;
      }
68    /**
       * Actualiza la tarea en el medio de almacenamiento
70     * @param tarea
       * @return
72     */
      public Tarea actualizarTarea(Tarea tarea){
74      Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
        session.beginTransaction();
76      session.update(tarea);
```

```
    session.getTransaction().commit();
78    return tarea;
  }
80
  public Tarea eliminarTarea(IdTarea idTarea){
82    Session session = HibernateUtil.getSessionFactory().
          getCurrentSession();
      session.beginTransaction();
84    Tarea tarea = null;
      try{
86      tarea = (Tarea)session.load(Tarea.class, idTarea);
        if( tarea != null){
88        session.delete(tarea);
        }
90    }catch(Exception e){
        e.printStackTrace();
92    }
      session.getTransaction().commit();
94    return tarea;
  }
96
  /**
98 * Obtiene la lista de tareas que predecen a Tarea
   *
100 * @param idUsuario
   * @return
102 */
  public Tarea cargarTareasPrecedentes(Tarea tarea){
104   Session session = HibernateUtil.getSessionFactory().
          getCurrentSession();
      session.beginTransaction();
106   try {
        /**
108      * utilizamos load porque ya sabemos que existe la tarea
         */
110     tarea = (Tarea)session.load(Tarea.class, tarea.getIdTarea());
        /**
112      * realizamos una operacion para que se carguen las tareas
             precedentes.
         */
114     tarea.getPredecesores().size();
      } catch (HibernateException e) {
116     e.printStackTrace();
        session.getTransaction().rollback();
118   }
      session.getTransaction().commit();
120   return tarea;
  }
122
  /**
124 * Obtiene la lista de usuarios de esta tarea
   *
```

```
126     * @param tarea
        * @return
128     */
    public Tarea cargarUsuariosTareasPrecedentes(Tarea tarea){
130     Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
        session.beginTransaction();
132     try {
          /**
134        * utilizamos load porque ya sabemos que existe la tarea
           */
136       tarea = (Tarea)session.load(Tarea.class, tarea.getIdTarea());
          /**
138        * realizamos una operacion para que se carguen las tareas
               precedentes.
           */
140       tarea.getRecursos().size();
          tarea.getPredecesores().size();
142       tarea.getSubtareas().size();
        } catch (HibernateException e) {
144       e.printStackTrace();
          session.getTransaction().rollback();
146     }
        session.getTransaction().commit();
148     return tarea;
    }
150
    /**
152  * Obtiene el num de tarea ultimo asignado
     */
154 public long numTareaMax(long idProyecto, long numIteracion){
      Session session = HibernateUtil.getSessionFactory().
          getCurrentSession();
156   session.beginTransaction();
      Query query;
158   long numTareaMax = 0;
      List<Integer> id;
160   try{
        query = session.createSQLQuery("SELECT MAX(numTarea) FROM Tarea
            WHERE Iteracion_Proyecto_idProyecto=? and
            Iteracion_numIteracion=?");
162     query.setString(0, idProyecto+"");
        query.setString(1, numIteracion+"");
164     id = (List<Integer>)query.list();
        numTareaMax = (Integer)id.get(0);
166   }catch(Exception e){
        e.printStackTrace();
168     numTareaMax = 0;
      }
170   session.getTransaction().commit();
      return numTareaMax;
172 }
```

```
  /**
   * Crea una lista de las subtareas de una tarea.
   * @param id
   * @return
   */
  public List<Tarea> subtareasDeTarea(IdTarea id){
    List<Tarea> subtareas=null;
    Session session = HibernateUtil.getSessionFactory().
        getCurrentSession();
    session.beginTransaction();
    SQLQuery query;
    try{
      query = session.createSQLQuery("select * from Tarea inner join (
          select numTareaHijo, numIteracionH, numProyectoH from
          PadreTareaHijo where numTarea=? and numIteracion=? and
          numProyecto=?)as b where b.numTareaHijo=Tarea.numTarea and b.
          numProyectoH=Tarea.Iteracion_Proyecto_idProyecto and b.
          numIteracionH=Tarea.Iteracion_numIteracion;");
      query.setString(0, id.getNumTarea()+"");
      query.setString(1, id.getNumIteracion()+"");
      query.setString(2, id.getIdProyecto()+"");
      query.addEntity(Tarea.class);
      subtareas = (List<Tarea>)query.list();
    }catch(HibernateException e){
      e.printStackTrace();
      session.getTransaction().rollback();
    }
    session.getTransaction().commit();
    return subtareas;
  }
  /**
   * Carga la lista de subtareas
   * @param id
   * @return
   */
  public Tarea cargarSubtareas(Tarea tarea){
    Session session = HibernateUtil.getSessionFactory().
        getCurrentSession();
    session.beginTransaction();
    try{
      tarea = (Tarea)session.load(Tarea.class, tarea.getIdTarea());
      tarea.getSubtareas().size();
    }catch(HibernateException e){
      e.printStackTrace();
      session.getTransaction().rollback();
    }
    session.getTransaction().commit();
    return tarea;
  }
}
```

src/com/gestion/modelo/manejadorBd/fase/ManejadorTarea.java

```
1  package com.gestion.modelo.manejadorBd.fase;
2
3  import java.util.List;
4
5  import org.hibernate.HibernateException;
6  import org.hibernate.Query;
7  import org.hibernate.SQLQuery;
8  import org.hibernate.classic.Session;
9  import org.hibernate.mapping.Array;
10
11 import com.gestion.modelo.proyecto.IdIteracion;
12 import com.gestion.modelo.proyecto.Iteracion;
13 import com.gestion.modelo.proyecto.Proyecto;
14 import com.gestion.modelo.proyecto.Tarea;
15 import com.gestion.util.HibernateUtil;
16
17 public class ManejadorIteracion {
18
19   public List<Iteracion> lista(){
       Session session = HibernateUtil.getSessionFactory().
           getCurrentSession();
21     session.beginTransaction();
       List<Iteracion> lista = null;
23     try{
         lista = (List<Iteracion>)session.createQuery("FROM Iteracion").
             list();
25     }catch(Exception e){
         e.printStackTrace();
27     }
       session.getTransaction().commit();
29     return lista;
     }
31
     /**
33    * Agrega una nueva iteracion
      * @param iteracion
35    * @return
      */
37   public Iteracion agregarIteracion(Iteracion iteracion){
       Session session = HibernateUtil.getSessionFactory().
           getCurrentSession();
39     session.beginTransaction();
       session.save(iteracion);
41     session.getTransaction().commit();
       return iteracion;
43   }
     /**
45    * Actualiza la iteracion
      * @param iteracion
47    * @return
      */
```

```
49    public Iteracion actualizarIteracion(Iteracion iteracion){
        Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
51      session.beginTransaction();
        try{
53        session.update(iteracion);
        }catch(Exception e){
55        e.printStackTrace();
          session.getTransaction().rollback();
57      }
        session.getTransaction().commit();
59      return iteracion;
      }
61    /**
       * Elimina iteracion de la base de datos.
63     * @param iteracion
       * @return
65     */
      public Iteracion eliminarIteracion(Iteracion iteracion){
67      Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
        session.beginTransaction();
69      session.delete(iteracion);
        session.getTransaction().commit();
71      return iteracion;
      }
73
      /**
75     * Eliminar la iteracion con identificador id
       * @param id
77     * @return
       */
79    public Iteracion eliminarIteracion(IdIteracion id){
        Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
81      session.beginTransaction();
        Iteracion iteracion = (Iteracion) session.get(Iteracion.class, id);
83      if(null != iteracion) {
          session.delete(iteracion);
85        System.out.println("Se elimino la iteracion: "+iteracion.getNombre
              ());
        }
87      session.getTransaction().commit();
        return iteracion;
89    }
      /**
91     * Busca una iteracion con el identificador 'id'.
       * @param id
93     * @return
       */
95    public Iteracion buscarIteracion(IdIteracion id){
```

```
    Session session = HibernateUtil.getSessionFactory().
        getCurrentSession();
97  session.beginTransaction();
    Iteracion iteracion=null;
99  try{
      iteracion = (Iteracion)session.get(Iteracion.class, id);
101 }catch(Exception e){
      e.printStackTrace();
103 }
    session.getTransaction().commit();
105 return iteracion;
  }
107
  /**
109  * Obtiene la lista de iteraciones del iteracion con identificador '
       idProyecto'
   * @param idProyecto Identificador del proyecto.
111  * @return Las
   */
113 public List<Iteracion> iteracionesProyecto(long idProyecto){
    Session session = HibernateUtil.getSessionFactory().
        getCurrentSession();
115 session.beginTransaction();
    List<Iteracion> iteraciones= null;
117 Proyecto p = null;
    Query query;
119 try {
      p = (Proyecto)session.get(Proyecto.class, idProyecto);
121   System.out.println("El nombre del proyecto es: "+p.getNombre());
      System.out.println(p.getIteraciones().size());
123   query = session.createSQLQuery("SELECT * FROM Iteracion WHERE
          Proyecto_idProyecto=?").addEntity(Iteracion.class);
      iteraciones = p.getIteraciones();
125   System.out.println("El numero de iteraciones son: "+iteraciones.
          size());
    } catch (HibernateException e){
127   e.printStackTrace();
      session.getTransaction().rollback();
129 }
    session.getTransaction().commit();
131 return iteraciones;
  }
133 /**
   * ESTE METODO NO ME FUNCIONO (16 DE JUNIO 4:03 AM.)
135  * Obtiene las tareas de la iteracion dada de la disciplina dada
   * @param iteracion
137  * @return
   */
139 public List<Tarea> tareasIteracionxxxx(IdIteracion id, long
      idDisciplina){
    Session session = HibernateUtil.getSessionFactory().
        getCurrentSession();
```

```
141     session.beginTransaction();
        List<Tarea> tareas= null;
143     SQLQuery query;
        try {
145       query = session.createSQLQuery("select  numTarea,
              Iteracion_Proyecto_idProyecto, Iteracion_numIteracion,
              Actividad_idActividad, nombreTarea, fechaInicio, fechaFin,
              hito, porcentaje, pGroup, pOpen, pCaption, indice from Tarea
              inner join (select idActividad from Actividad inner join
              Disciplina where Actividad.Disciplina_idDisciplina=Disciplina.
              idDisciplina and Disciplina.idDisciplina=1) as res1 where
              Tarea.Actividad_idActividad=res1.idActividad;");
          query.addEntity(Tarea.class);
147       tareas = (List<Tarea>)query.list();
        } catch (HibernateException e){
149       e.printStackTrace();
          session.getTransaction().rollback();
151     }
        session.getTransaction().commit();
153     return tareas;
    }
155

157 /**
     * Obtiene las tareas de la iteracion dada.
159  * @param iteracion
     * @return
161  */
    public List<Tarea> tareasIteracion(IdIteracion id){
163   Session session = HibernateUtil.getSessionFactory().
          getCurrentSession();
      session.beginTransaction();
165   List<Tarea> tareas= null;
      Iteracion iter;
167   try {
        iter = (Iteracion)session.get(Iteracion.class, id);
169     iter.getTareas().size();
        tareas = iter.getTareas();
171   } catch (HibernateException e){
        e.printStackTrace();
173     session.getTransaction().rollback();
      }
175   session.getTransaction().commit();
      return tareas;
177 }
    /**
179  * Obtiene el ultimo numero asignado a un iteracion
     * para el proyecto.
181  * @param idProyecto Identificar del proyecto.
     * @return
183  */
    public Integer getMaxNumIteracion(long idProyecto){
```

```
185     Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
        session.beginTransaction();
187     Integer max=0;
        Query query;
189     try {
          query = session.createSQLQuery("SELECT MAX(numIteracion) FROM
              Iteracion WHERE Proyecto_idProyecto=?");
191       query.setString(0, idProyecto+"");
          List<?> result = query.list();
193       max = (Integer) result.get(0);
        } catch (HibernateException e){
195       e.printStackTrace();
          session.getTransaction().rollback();
197     }
        session.getTransaction().commit();
199     return max;
    }
201
    /**
203  * Carga las tareas asociadas a la iteracion.
     * @param iteracion
205  * @return
     */
207 public Iteracion cargarTareas(Iteracion iteracion){
      Session session = HibernateUtil.getSessionFactory().
          getCurrentSession();
209   session.beginTransaction();
      try{
211     iteracion = (Iteracion)session.load(Iteracion.class, iteracion.
            getId());
        iteracion.getTareas().size();
213   }catch(Exception e){
        e.printStackTrace();
215   }
      session.getTransaction().commit();
217   return iteracion;
    }
219
}
```

src/com/gestion/modelo/manejadorBd/fase/ManejadorIteracion.java

```
  package com.gestion.modelo.manejadorBd.fase;
2
  import java.util.List;
4
  import org.hibernate.HibernateException;
6 import org.hibernate.classic.Session;

8 import com.gestion.modelo.proyecto.Fase;
  import com.gestion.modelo.proyecto.Proyecto;
10 import com.gestion.util.HibernateUtil;
```

```
12 public class ManejadorFase {
     /**
14    * Recupera las fases existentes son solo 4, aplicables
      * para cualquier proyecto.
16    * @return
      */
18   public List<Fase> listaFases(){
       Session session = HibernateUtil.getSessionFactory().
           getCurrentSession();
20     session.beginTransaction();
       List<Fase> fases = null;
22     try {
         fases = (List<Fase>)session.createQuery("from Fase").list();
24     } catch (HibernateException e) {
         e.printStackTrace();
26       session.getTransaction().rollback();
       }
28     session.getTransaction().commit();
       return fases;
30   }
     /**
32    * Recupera la fase que tiene el identificador 'idFase'
      * @param idFase Identificador de la fase
34    * @return Objeto Fase
      */
36   public Fase fase(long idFase){
       Session session = HibernateUtil.getSessionFactory().
           getCurrentSession();
38     session.beginTransaction();
       Fase fase= (Fase)session.get(Fase.class, idFase);
40     session.getTransaction().commit();
       return fase;
42   }
     /**
44    * Almacena una Fase en la base de datos.
      * @param fase
46    * @return
      */
48   public Fase agregarFase(Fase fase){
       Session session = HibernateUtil.getSessionFactory().
           getCurrentSession();
50     session.beginTransaction();
       session.save(fase);
52     session.getTransaction().commit();
       return fase;
54   }
     /**
56    * Actualizar la fase
      * @param fase
58    * @return
      */
```

```
60    public Fase actualizarFase(Fase fase){
        Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
62      session.beginTransaction();
        session.update(fase);
64      session.getTransaction().commit();
        return fase;
66    }
      /**
68     * Elimina la fase
       * @param fase
70     * @return
       */
72    public Fase eliminarFase(Fase fase){
        Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
74      session.beginTransaction();
        session.delete(fase);
76      session.getTransaction().commit();
        return fase;
78    }
  }
```

src/com/gestion/modelo/manejadorBd/fase/ManejadorFase.java

```
1  package com.gestion.modelo.manejadorBd.fase;

3  import java.util.List;

5  import org.hibernate.Session;

7  import com.gestion.modelo.proyecto.Disciplina;
   import com.gestion.util.HibernateUtil;
9
   public class ManejadorDisciplina {
11   /**
      * Recupera todas las instancias de disciplina disponibles
13    * en esl sistema
      * @return
15    */
     public List<Disciplina> lista(){
17     Session session = HibernateUtil.getSessionFactory().
           getCurrentSession();
       session.beginTransaction();
19     List<Disciplina> disciplinas = null;
       try{
21       disciplinas = (List<Disciplina>)session.createQuery("FROM
             Disciplina").list();
       }catch(Exception e){
23       e.printStackTrace();
       }
25     session.getTransaction().commit();
       return disciplinas;
```

```
27    }
      /**
29     * Agrega una disciplina en la base de datos.
       * @param disciplina
31     * @return
       */
33    public Disciplina agregarDisciplina(Disciplina disciplina){
        Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
35      session.beginTransaction();
        session.save(disciplina);
37      session.getTransaction().commit();
        return disciplina;
39    }
      /**
41     * Actualiza la disciplina en la base de datos.
       * @param disciplina
43     * @return
       */
45    public Disciplina actualizarDisciplina(Disciplina disciplina){
        Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
47      session.beginTransaction();
        session.update(disciplina);
49      session.getTransaction().commit();
        return disciplina;
51    }
      /**
53     * Elimina la disciplina de la base de datos.
       * @param disciplina
55     * @return
       */
57    public Disciplina eliminarDisciplina(Disciplina disciplina){
        Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
59      session.beginTransaction();
        session.delete(disciplina);
61      session.getTransaction().commit();
        return disciplina;
63    }
      /**
65     * Busca la disciplina con el id dado.
       * @param idDisciplina
67     * @return
       */
69    public Disciplina buscar(long idDisciplina){
        Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
71      session.beginTransaction();
        Disciplina disciplina = null;
73      try{
```

```
      disciplina = (Disciplina)session.get(Disciplina.class,idDisciplina
          );
    }catch(Exception e){
      e.printStackTrace();
      session.getTransaction().rollback();
      return null;
    }
    session.getTransaction().commit();
    return disciplina;
  }
}
```

src/com/gestion/modelo/manejadorBd/fase/ManejadorDisciplina.java

```
package com.gestion.modelo.manejadorBd.artefacto;

import java.util.List;
import java.util.Set;

import org.hibernate.classic.Session;

import com.gestion.modelo.proyecto.Artefacto;
import com.gestion.modelo.rol.Rol;
import com.gestion.util.HibernateUtil;

public class ManejadorArtefacto {
  public List<Artefacto> lista(){
    Session session = new HibernateUtil().getSessionFactory().
        getCurrentSession();
    session.beginTransaction();
    List<Artefacto> artefactos = null;
    try{
      artefactos = (List<Artefacto>)session.createQuery("from Artefacto"
          ).list();
    }catch(Exception e){
      e.printStackTrace();
      session.getTransaction().rollback();
    }
    session.getTransaction().commit();
    return artefactos;
  }
  /**
   * Agregar un Artefacto
   * @param Artefacto
   * @return
   */
  public Artefacto add(Artefacto artefacto){
    Session session = HibernateUtil.getSessionFactory().
        getCurrentSession();
    session.beginTransaction();
    session.save(artefacto);
    session.getTransaction().commit();
    return artefacto;
```

```
37    }
      /**
39     * Buscar un Artefacto con el nombre dado.
       * @param nombre_Artefacto
41     * @return
       */
43    public Artefacto find(String nombre_artefacto){
        Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
45      session.beginTransaction();
        Artefacto Artefacto = (Artefacto)session.get(Artefacto.class,
            nombre_artefacto);
47      session.getTransaction().commit();
        return Artefacto;
49    }
      /**
51     * Actualizar el Artefacto especificado
       * @param Artefacto
53     * @return
       */
55    public Artefacto actualizarArtefacto(Artefacto artefacto){
        Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
57      session.beginTransaction();
        session.update(artefacto);
59      session.getTransaction().commit();
        return artefacto;
61    }
      /**
63     * Elimina de la base de datos el Artefacto especificado
       * @param Artefacto
65     * @return
       */
67    public Artefacto eliminarArtefacto(Artefacto artefacto){
        Session session = HibernateUtil.getSessionFactory().
            getCurrentSession();
69      session.beginTransaction();
        session.delete(artefacto);
71      session.getTransaction().commit();
        return artefacto;
73    }

75  }
```

src/com/gestion/modelo/manejadorBd/artefacto/ManejadorArtefacto.java

```
1   package com.gestion.modelo.manejadorBd.ejemplarArtefacto;

3   import java.util.List;

5   import org.hibernate.HibernateException;
    import org.hibernate.classic.Session;
7
```

```
import com.gestion.modelo.ejemplarArtefacto.EjemplarArtefacto;
import com.gestion.modelo.ejemplarArtefacto.IdEjemplarArtefacto;
import com.gestion.util.HibernateUtil;

public class ManejadorEjemplarArtefacto {
  public List<EjemplarArtefacto> lista(){
    Session session = new HibernateUtil().getSessionFactory().
        getCurrentSession();
    session.beginTransaction();
    List<EjemplarArtefacto> artefactos = null;
    try{
      artefactos = (List<EjemplarArtefacto>)session.createQuery("from
          EjemplarArtefacto").list();
    }catch(Exception e){
      e.printStackTrace();
      session.getTransaction().rollback();
    }
    session.getTransaction().commit();
    return artefactos;
  }
  /**
   * Agregar un Artefacto
   * @param Artefacto
   * @return
   */
  public EjemplarArtefacto add(EjemplarArtefacto artefacto){
    Session session = HibernateUtil.getSessionFactory().
        getCurrentSession();
    session.beginTransaction();
    session.saveOrUpdate(artefacto);
    session.getTransaction().commit();
    System.out.println("Operacion de guardado efectuado");
    return artefacto;
  }

  public EjemplarArtefacto buscar(IdEjemplarArtefacto idEjemplar){
    Session session = HibernateUtil.getSessionFactory().
        getCurrentSession();
    session.beginTransaction();
    EjemplarArtefacto ejemplar = null;
    try{
      ejemplar = (EjemplarArtefacto)session.get(EjemplarArtefacto.class,
          idEjemplar);
    }catch(HibernateException e){
      e.printStackTrace();
      session.getTransaction().rollback();
      return null;
    }
    session.getTransaction().commit();
    return ejemplar;

  }
```

```
55 }
```

src/com/gestion/modelo/manejadorBd/ejemplarArtefacto/ManejadorEjemplarArtefacto.java

```
1  package com.gestion.modelo.usuario;

3  import java.io.Serializable;

5  import javax.persistence.EmbeddedId;
   import javax.persistence.Entity;
7  import javax.persistence.JoinColumn;
   import javax.persistence.OneToOne;
9
   import com.gestion.modelo.rol.Rol;
11
   @Entity
13 public class Permiso implements Serializable{
     private IdPermiso id;
15   private Accion accion;
     private Rol rol;
17   private Boolean permitido;

19   public Permiso(){

21   }

23   @EmbeddedId
     public IdPermiso getId() {
25     return id;
     }
27   public void setId(IdPermiso id) {
       this.id = id;
29   }
     @OneToOne
31   @JoinColumn(name="idAccion",insertable=false,updatable=false)
     public Accion getAccion() {
33     return accion;
     }
35   public void setAccion(Accion accion) {
       this.accion = accion;
37   }
     @OneToOne
39   @JoinColumn(name="rol",insertable=false,updatable=false)
     public Rol getRol() {
41     return rol;
     }
43   public void setRol(Rol rol) {
       this.rol = rol;
45   }
     public Boolean getPermitido() {
47     return permitido;
     }
49   public void setPermitido(Boolean permitido) {
```

```
    this.permitido = permitido;
  }


}
```

src/com/gestion/modelo/usuario/Permiso.java

```
package com.gestion.modelo.usuario;

import java.io.Serializable;

import javax.persistence.Embeddable;

@Embeddable
public class IdPermiso implements Serializable {
  private Integer idAccion;
  private String rol;

  public IdPermiso(){}

  public Integer getIdAccion() {
    return idAccion;
  }
  public void setIdAccion(Integer idAccion) {
    this.idAccion = idAccion;
  }
  public String getRol() {
    return rol;
  }
  public void setRol(String rol) {
    this.rol = rol;
  }
}
```

src/com/gestion/modelo/usuario/IdPermiso.java

```
package com.gestion.modelo.usuario;

import java.io.Serializable;
import java.util.ArrayList;
import java.util.HashSet;
import java.util.List;
import java.util.Set;

import javax.persistence.*;

import com.gestion.modelo.proyecto.Proyecto;
import com.gestion.modelo.proyecto.Tarea;
import com.gestion.modelo.rol.Rol;

@Entity
public class Usuario implements Serializable{
```

```
  private String idUsuario;
18  private String nombre;
  private String ape_paterno;
20  private String ape_materno;
  private String correo;
22  private String telefono;
  private String login;
24  private String pass_word;
  private Set<Rol> roles= new HashSet<Rol>();
26  private List<Proyecto> proyectos = new ArrayList<Proyecto>();
  private String nombreCompleto;
28  private List<Tarea> tareas;

30

  public Usuario(){

32

  }

34

  @Id
36  @Column(name="CURP")
  public String getIdUsuario() {
38    return idUsuario;
  }

40

  public void setIdUsuario(String idPersonal) {
42    this.idUsuario = idPersonal;
  }

44

  public String getNombre() {
46    return nombre;
  }

48

  public void setNombre(String nombre) {
50    this.nombre = nombre;
  }

52

  public String getApe_paterno() {
54    return ape_paterno;
  }

56

  public void setApe_paterno(String apePaterno) {
58    ape_paterno = apePaterno;
  }

60

  public String getApe_materno() {
62    return ape_materno;
  }

64

  public void setApe_materno(String apeMaterno) {
66    ape_materno = apeMaterno;
  }

68
```

```
  public String getCorreo() {
70  return correo;
  }
72
  public void setCorreo(String correo) {
74  this.correo = correo;
  }
76
  public String getTelefono() {
78  return telefono;
  }
80
  public void setTelefono(String telefono) {
82  this.telefono = telefono;
  }
84
  public String getLogin() {
86  return login;
  }
88
  public void setLogin(String login) {
90  this.login = login;
  }
92
  public String getPass_word() {
94  return pass_word;
  }
96
  public void setPass_word(String passWord) {
98  pass_word = passWord;
  }
100
  @ManyToMany(targetEntity=Rol.class)
102 @JoinTable(name="UsuarioRol",joinColumns=@JoinColumn(name="
      Usuario_CURP"),
       inverseJoinColumns = @JoinColumn( name="Rol_nombreRol"))
104 public Set<Rol> getRoles() {
  return roles;
106 }

108 public void setRoles(Set<Rol> roles) {
  this.roles = roles;
110 }

112 public void addRol(Rol rol){
  this.roles.add(rol);
114 }
  @Transient
116 public String getNombreCompleto() {
  return this.nombre + " "+ape_paterno+" "+ape_materno;
118 }
```

```
120   public void setNombreCompleto(String nombreCompleto) {
        this.nombreCompleto = nombreCompleto;
122   }

124   @ManyToMany(mappedBy="listaUsuarios")
      public List<Proyecto> getProyectos() {
126     return proyectos;
      }
128
      public void setProyectos(List<Proyecto> proyectos) {
130     this.proyectos = proyectos;
      }
132   @ManyToMany(mappedBy="recursos")
      public List<Tarea> getTareas() {
134     return tareas;
      }
136
      public void setTareas(List<Tarea> tareas) {
138     this.tareas = tareas;
      }
140
      @Override
142   public boolean equals(Object other){
          if (this == other) return true;
144
          if ( !(other instanceof Usuario) ) return false;
146
          final Usuario u = (Usuario) other;
148
          if( u.getIdUsuario().compareTo(this.getIdUsuario())==0)
150         return true;
          else return false;
152   }


154
      @Override
156   public int hashCode() {
        final int prime = 31;
158     int result = 1;
        result = prime * result
160         + ((idUsuario == null) ? 0 : idUsuario.hashCode());
        return result;
162   }
}
```

src/com/gestion/modelo/usuario/Usuario.java

```
1 package com.gestion.modelo.usuario;

3 import java.io.Serializable;

5 import javax.persistence.Entity;
  import javax.persistence.Id;
```

```
7
  @Entity
9 public class Accion implements Serializable{
    private Integer id;
11  private String nombre;
    private String descripcion;
13  private String categoria;

15  public Accion(){

17  }


19


21  @Id
    public Integer getId() {
23    return id;
    }
25
    public void setId(Integer id) {
27    this.id = id;
    }
29  public String getNombre() {
      return nombre;
31  }
    public void setNombre(String nombre) {
33    this.nombre = nombre;
    }
35
    public String getDescripcion() {
37    return descripcion;
    }
39  public void setDescripcion(String descripcion) {
      this.descripcion = descripcion;
41  }
    public String getCategoria() {
43    return categoria;
    }
45  public void setCategoria(String categoria) {
      this.categoria = categoria;
47  }

49 }
```

src/com/gestion/modelo/usuario/Accion.java

```
1 package com.gestion.modelo.ejemplarArtefacto;

3 import java.io.Serializable;
  import java.sql.Blob;
5 import java.sql.Date;

7 import javax.persistence.EmbeddedId;
```

```
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.Lob;
import javax.persistence.OneToOne;

import com.gestion.modelo.proyecto.Artefacto;
import com.gestion.modelo.proyecto.Proyecto;

@Entity
public class EjemplarArtefacto implements Serializable{
  private IdEjemplarArtefacto idEjemplar;
  private String nombreArchivo;
  private String tipoContenido;
  private Artefacto artefacto;
  private Blob contenidoArchivo;
  private Proyecto proyecto;
  private Date fechaCreacion;

  public EjemplarArtefacto(){

  }
  @EmbeddedId
  public IdEjemplarArtefacto getIdEjemplar() {
    return idEjemplar;
  }

  public void setIdEjemplar(IdEjemplarArtefacto idEjemplar) {
    this.idEjemplar = idEjemplar;
  }

  public String getNombreArchivo() {
    return nombreArchivo;
  }
  public void setNombreArchivo(String nombreArchivo) {
    this.nombreArchivo = nombreArchivo;
  }

  public String getTipoContenido() {
    return tipoContenido;
  }
  public void setTipoContenido(String tipoContenido) {
    this.tipoContenido = tipoContenido;
  }
  @OneToOne
  @JoinColumn(name="Artefacto_nombre",insertable=false, updatable=false)
  public Artefacto getArtefacto() {
    return artefacto;
  }
  public void setArtefacto(Artefacto artefacto) {
    this.artefacto = artefacto;
  }
```

```
  @Lob
61  public Blob getContenidoArchivo() {
    return contenidoArchivo;
63  }
  public void setContenidoArchivo(Blob contenidoArchivo) {
65    this.contenidoArchivo = contenidoArchivo;
  }
67  @OneToOne
  @JoinColumn(name="Proyecto_idProyecto",insertable=false, updatable=
      false)
69  public Proyecto getProyecto() {
    return proyecto;
71  }
  public void setProyecto(Proyecto proyecto) {
73    this.proyecto = proyecto;
  }
75  public Date getFechaCreacion() {
    return fechaCreacion;
77  }
  public void setFechaCreacion(Date fechaCreacion) {
79    this.fechaCreacion = fechaCreacion;
  }
81}
```

src/com/gestion/modelo/ejemplarArtefacto/EjemplarArtefacto.java

```
1 package com.gestion.modelo.ejemplarArtefacto;

3 import java.io.Serializable;

5 import javax.persistence.Column;

7 public class IdEjemplarArtefacto implements Serializable {
    private long idProyecto;
9   private String nombreArtefacto;

11  public IdEjemplarArtefacto(long idProyecto, String nombreArtefacto){
      this.idProyecto = idProyecto;
13    this.nombreArtefacto = nombreArtefacto;
    }
15


17  @Column(name="Proyecto_idProyecto")
    public long getIdProyecto() {
19    return idProyecto;
    }
21  public void setIdProyecto(long idProyecto) {
      this.idProyecto = idProyecto;
23  }
    @Column(name="Artefacto_nombre")
25  public String getNombreArtefacto() {
      return nombreArtefacto;
27  }
```

```
  public void setNombreArtefacto(String nombreArtefacto) {
29    this.nombreArtefacto = nombreArtefacto;
  }
31}
```

src/com/gestion/modelo/ejemplarArtefacto/IdEjemplarArtefacto.java

```
 1package com.gestion.util;

 3public class PermisoDTO {
  private String rol;
 5  private Integer idAccion;
  private String accion;
 7  private String descripcionAccion;
  private String categoria;
 9  private Boolean permitido;

11  public PermisoDTO(){

13  }

15  public PermisoDTO(Object idAccion, Object accion, Object descripcion,
      Object categoria,Object rol,  Object permitido){
    this.rol = (String)rol;
17    this.idAccion = (Integer)idAccion;
    this.accion = (String)accion;
19    this.descripcionAccion = (String)descripcion;
    this.categoria = (String)categoria;
21    this.permitido = (Boolean)permitido;
  }
23
  public String getRol() {
25    return rol;
  }
27  public void setRol(String rol) {
    this.rol = rol;
29  }
  public Integer getIdAccion() {
31    return idAccion;
  }
33  public void setIdAccion(Integer idAccion) {
    this.idAccion = idAccion;
35  }
  public String getAccion() {
37    return accion;
  }
39  public void setAccion(String accion) {
    this.accion = accion;
41  }
  public String getDescripcionAccion() {
43    return descripcionAccion;
  }
45  public void setDescripcionAccion(String descripcionAccion) {
```

```
    this.descripcionAccion = descripcionAccion;
  }
  public String getCategoria() {
    return categoria;
  }
  public void setCategoria(String categoria) {
    this.categoria = categoria;
  }
  public Boolean getPermitido() {
    return permitido;
  }
  public void setPermitido(Boolean permitido) {
    this.permitido = permitido;
  }

  public String getPermitidoS(){
    if(permitido!=null && permitido==true){
      return "Si";
    }else{
      return "No";
    }
  }


}
```

src/com/gestion/util/PermisoDTO.java

```
package com.gestion.util;

import org.hibernate.SessionFactory;
import org.hibernate.cfg.AnnotationConfiguration;

public class HibernateUtil {

  private static final SessionFactory sessionFactory =
      buildSessionFactory();

  private static SessionFactory buildSessionFactory() {
    try {
      // Create the SessionFactory from hibernate.cfg.xml
      return new AnnotationConfiguration().configure()
          .buildSessionFactory();
    } catch (Throwable ex) {
      System.err.println("Initial SessionFactory creation failed." + ex)
          ;
      throw new ExceptionInInitializerError(ex);
    }
  }

  public static SessionFactory getSessionFactory() {
    return sessionFactory;
  }
```

```
24 }
```

src/com/gestion/util/HibernateUtil.java

```
package com.gestion.vista.json;

import java.util.HashSet;
import java.util.Set;

import com.gestion.modelo.rol.Rol;
import com.gestion.modelo.usuario.Usuario;

public class UsuarioJson {
  private String idUsuario;
  private String nombreCompleto;
  public UsuarioJson(){

  }
  public UsuarioJson(Usuario u){
    this.idUsuario = u.getIdUsuario();
    this.nombreCompleto = u.getNombreCompleto();
  }
  public String getIdUsuario() {
    return idUsuario;
  }
  public void setIdUsuario(String idUsuario) {
    this.idUsuario = idUsuario;
  }
  public String getNombreCompleto() {
    return nombreCompleto;
  }
  public void setNombreCompleto(String nombreCompleto) {
    this.nombreCompleto = nombreCompleto;
  }

}
```

src/com/gestion/vista/json/UsuarioJson.java

```
package com.gestion.vista.json;

public class Libro {
  private String nombre;
  private String autor;

  public Libro(String nombre, String autor){
    this.nombre = nombre;
    this.autor = autor;
  }

  public String getNombre() {
    return nombre;
  }
```

```
16    public void setNombre(String nombre) {
        this.nombre = nombre;
18    }

20    public String getAutor() {
        return autor;
22    }

24    public void setAutor(String autor) {
        this.autor = autor;
26    }

28  }
```

src/com/gestion/vista/json/Libro.java

```
  package com.gestion.vista.json;
2
  import java.util.ArrayList;
4
  import java.util.Date;
6 import java.util.Iterator;
  import java.util.List;
8
  import com.gestion.modelo.proyecto.Disciplina;
10 import com.gestion.modelo.proyecto.Tarea;
  import com.gestion.modelo.usuario.Usuario;
12


14 public class TareaJson implements Comparable{
    private long idProyecto;
16  private long numIteracion;
    private long numTarea;
18  /*
     * un clave que se genera concatenando
20   * idProyecto + numIteracion + idTarea
     */
22  private long idTarea;
    private String nombreTarea;
24  private String fechaInicio;
    private String fechaFin;
26  private int hito;
    private int porcentaje;
28  private long pGroup;
    private long tareaPadre;
30  private long pOpen;
    private String pCaption;
32  private String predecesores = "";
    private String recursos="";
34  public TareaJson(Tarea tarea){
      System.out.println("PGROUP: "+tarea.getpGroup());
36    this.idProyecto = tarea.getIdTarea().getIdProyecto();
```

```
    this.numIteracion = tarea.getIdTarea().getNumIteracion();
38  this.numTarea = tarea.getIdTarea().getNumTarea();
    this.idTarea = Integer.valueOf(tarea.getIdTarea().toString());
40  this.nombreTarea = tarea.getNombreTarea();
    this.fechaInicio = tarea.getFechaInicio().toString();
42  this.fechaFin = tarea.getFechaFin().toString();
    this.hito = tarea.getHito();
44  this.porcentaje = tarea.getPorcentaje();

46  if( tarea.getTareaPadre().size()>0){
      this.tareaPadre = Integer.valueOf(tarea.getTareaPadre().get(0).
          getIdTarea().toString());
48  }else{
      this.tareaPadre = tarea.getActividad().getDisciplina().
          getIdDisciplina();
50  }
    if( tarea.getSubtareas().size()>0){
52    this.pOpen = 1;
      this.pGroup = 1;
54  }else{
      this.pOpen = tarea.getpOpen();
56    this.pGroup = tarea.getpGroup();
    }
58  this.pCaption = tarea.getpCaption();
    List<String> lista = new ArrayList<String>();
60  Iterator<Tarea> it = tarea.getPredecesores().iterator();
    Tarea a=null;
62  while(it.hasNext()){
      a = it.next();
64    lista.add(a.getIdTarea()+"");
    }
66  String temp = lista.toString();
    if( temp.length()>2){
68    this.predecesores = temp.substring(1, temp.length()-1);
    }else{
70    this.predecesores = "";
    }
72
    Iterator<Usuario> it2 = tarea.getRecursos().iterator();
74  Usuario usr;
    lista = new ArrayList<String>();
76  while( it2.hasNext()){
      usr = it2.next();
78    lista.add(usr.getNombre());
    }
80  temp = lista.toString();
    if( temp.length()>2){
82    this.recursos = temp.substring(1, temp.length()-1);
    }else{
84    this.recursos = "";
    }
86 }
```

```
  public TareaJson(Disciplina d){
88    this.idTarea = d.getIdDisciplina();
      this.nombreTarea = d.getNombre();
90    this.fechaFin="";
      this.fechaInicio="";
92    this.hito =0;
      this.pCaption = d.getNombre();
94    this.pGroup = 1;
      this.pOpen = 0;
96    this.porcentaje = 0;
      this.predecesores = "";
98    this.recursos = "";
      this.tareaPadre = 0;
100 }

102 public long getIdTarea() {
      return idTarea;
104 }
    public void setIdTarea(long idTarea) {
106   this.idTarea = idTarea;
    }
108 public String getNombreTarea() {
      return nombreTarea;
110 }
    public void setNombreTarea(String nombreTarea) {
112   this.nombreTarea = nombreTarea;
    }
114 public String getFechaInicio() {
      return fechaInicio;
116 }
    public void setFechaInicio(String fechaInicio) {
118   this.fechaInicio = fechaInicio;
    }
120 public String getFechaFin() {
      return fechaFin;
122 }
    public void setFechaFin(String fechaFin) {
124   this.fechaFin = fechaFin;
    }
126 public int getHito() {
      return hito;
128 }
    public void setHito(int hito) {
130   this.hito = hito;
    }
132 public int getPorcentaje() {
      return porcentaje;
134 }
    public void setPorcentaje(int porcentaje) {
136   this.porcentaje = porcentaje;
    }
138 public long getpGroup() {
```

```
    return pGroup;
  }
  public void setpGroup(long pGroup) {
    this.pGroup = pGroup;
  }
  public long getTareaPadre() {
    return tareaPadre;
  }
  public void setTareaPadre(long tareaPadre) {
    this.tareaPadre = tareaPadre;
  }
  public long getpOpen() {
    return pOpen;
  }
  public void setpOpen(long pOpen) {
    this.pOpen = pOpen;
  }
  public String getpCaption() {
    return pCaption;
  }
  public void setpCaption(String pCaption) {
    this.pCaption = pCaption;
  }
  public String getPredecesores() {
    return predecesores;
  }
  public void setPredecesores(String predecesores) {
    this.predecesores = predecesores;
  }
  public String getRecursos() {
    return recursos;
  }
  public void setRecursos(String recursos) {
    this.recursos = recursos;
  }
  @Override
  public int compareTo(Object o) {
    // TODO Auto-generated method stub

    return 0;
  }
  public long getNumTarea() {
    return numTarea;
  }
  public void setNumTarea(long numTarea) {
    this.numTarea = numTarea;
  }
  public long getIdProyecto() {
    return idProyecto;
  }
  public void setIdProyecto(long idProyecto) {
    this.idProyecto = idProyecto;
```

```
  }
  public long getNumIteracion() {
    return numIteracion;
  }
  public void setNumIteracion(long numIteracion) {
    this.numIteracion = numIteracion;
  }

}
```

src/com/gestion/vista/json/TareaJson.java

```
package com.gestion.vista.json;

import com.gestion.modelo.proyecto.Actividad;


public class ActividadJson {
  private long idActividad;
  private String nombre;

  public ActividadJson(Actividad a){
    this.idActividad = a.getIdActividad();
    this.nombre = a.getNombre();
  }

  public long getIdActividad() {
    return idActividad;
  }
  public void setIdActividad(long idActividad) {
    this.idActividad = idActividad;
  }
  public String getNombre() {
    return nombre;
  }
  public void setNombre(String nombre) {
    this.nombre = nombre;
  }
}
```

src/com/gestion/vista/json/ActividadJson.java

```
package com.gestion.vista.json;

import java.util.ArrayList;
import java.util.Date;
import java.util.Iterator;
import java.util.List;

import com.gestion.modelo.manejadorBd.fase.ManejadorDisciplina;
import com.gestion.modelo.manejadorBd.fase.ManejadorTarea;
import com.gestion.modelo.proyecto.Disciplina;
import com.gestion.modelo.proyecto.Tarea;
```

```
import com.opensymphony.xwork2.Action;

public class JsonAction {
  private List<TareaJson> tareas2;
  private ManejadorTarea manejadorTareas;
  private ManejadorDisciplina manejadorDiscplina;

  public JsonAction(){
    manejadorTareas = new ManejadorTarea();
    manejadorDiscplina = new ManejadorDisciplina();
  }

  /**
   * Obtiene una lista de proyectos sobre los cuales
   * el usuario actual es jefe de proyecto.
   * @param idUsuario Identificador del usuario.
   * @return
   */
  public String proyectosUsuario(){
    try{
      List<Disciplina> disciplinas = manejadorDiscplina.lista();
      System.out.println("Hay "+ disciplinas.size()+" disciplinas");

      List<TareaJson> dis = new ArrayList<TareaJson>();

      Iterator<Disciplina> d = disciplinas.iterator();
      while(d.hasNext()){
        dis.add(new TareaJson(d.next()));
      }



      List<Tarea> tareas1;
        tareas1 = manejadorTareas.lista();
        Iterator<Tarea> it = tareas1.iterator();
        Tarea a;
        tareas2 = new ArrayList<TareaJson>();
        while(it.hasNext()){
          a = it.next();
          a = manejadorTareas.cargarUsuariosTareasPrecedentes(a);
          tareas2.add(new TareaJson(a));
        }
        tareas1 = null;
    }catch(Exception e){
      e.printStackTrace();
    }
        return Action.SUCCESS;
  }
    public String getJSON(){
      return proyectosUsuario();
    }
```

```
  public List<TareaJson> getTareas2() {
65  return tareas2;
  }
67
  public void setTareas2(List<TareaJson> tareas2) {
69  this.tareas2 = tareas2;
  }
71 }
```

src/com/gestion/vista/json/JsonAction.java

```
1 package com.gestion.vista.proyecto;

3 import java.util.Iterator;
  import java.util.List;
5 import java.util.Map;
  import java.util.Set;
7
  import org.apache.struts2.dispatcher.SessionMap;
9 import org.apache.struts2.interceptor.SessionAware;

11 import com.gestion.modelo.manejadorBd.proyecto.ManejadorProyecto;
  import com.gestion.modelo.manejadorBd.usuario.ManejadorUsuario;
13 import com.gestion.modelo.proyecto.Proyecto;
  import com.gestion.modelo.usuario.Usuario;
15 import com.opensymphony.xwork2.ActionSupport;

17 public class AdministrarProyectoAction extends ActionSupport implements
     SessionAware{
  private Usuario user;
19  private ManejadorProyecto manejadorProyecto;
  private ManejadorUsuario manejadorUsuario;
21  private List<Proyecto> proyectos;
  private List<Usuario> usuarios;
23
  private List<Usuario> jefes;
25  private String idJefe;

27  private Proyecto proyecto;
  private int idProyectoElegido;
29
  Map session;
31  public AdministrarProyectoAction() {
  manejadorProyecto = new ManejadorProyecto();
33  manejadorUsuario = new ManejadorUsuario();
  }
35  public List<Proyecto> getProyectos() {
  return proyectos;
37  }
  public void setProyectos(List<Proyecto> proyectos) {
39  this.proyectos = proyectos;
  }
41  public Proyecto getProyecto() {
```

```
    return proyecto;
43  }

45  public void setProyecto(Proyecto proyecto) {
    this.proyecto = proyecto;
47  }

49  public int getIdProyectoElegido() {
    return idProyectoElegido;
51  }

53  public void setIdProyectoElegido(int idProyectoElegido) {
    this.idProyectoElegido = idProyectoElegido;
55  }

57  public List<Usuario> getUsuarios() {
    return usuarios;
59  }

61  public void setUsuarios(List<Usuario> usuarios) {
    this.usuarios = usuarios;
63  }

65  public String getIdJefe() {
    return idJefe;
67  }

69  public void setIdJefe(String idJefe) {
    this.idJefe = idJefe;
71  }

73  public List<Usuario> getJefes() {
    return jefes;
75  }

77  public void setJefes(List<Usuario> jefes) {
    this.jefes = jefes;
79  }

81  /**
     * Obtiene una lista de proyectos sobre los cuales
83   * el usuario actual es jefe de proyecto o partipante del proyecto
     * Esta accion es llamada despues que el usuario se
85   * valido correctamente.
     * @param idUsuario Identificador del usuario.
87   * @return
     */
89  public String proyectosUsuario(){
    try{
91    String correo="";
      Usuario usr=null;
93    if (((Usuario)((SessionMap)session).get("usuario"))!=null){
```

```
            usr = (Usuario)((SessionMap)session).get("usuario");
95          correo = usr.getCorreo();
            usr = manejadorUsuario.usuario(correo);
97          usr = manejadorUsuario.cargarProyectos(usr);

99          List<Proyecto> p = manejadorProyecto.listaProyectos(usr.
                getIdUsuario());
            proyectos = usr.getProyectos();
101
            Iterator<Proyecto> it = p.iterator();
103         Proyecto p0;
            while(it.hasNext()){
105           p0 = it.next();
              if( !proyectos.contains(p0)){
107             proyectos.add(p0);
              }
109         }
        }
111     else{
          return LOGIN;
113     }
      }catch(Exception e){
115     e.printStackTrace();
        return LOGIN;
117   }
      return SUCCESS;
119 }
    /**
121  * Establecer
     * @return
123  */
    public String establecerProyectoActual(){
125   try{
        String res = proyectosUsuario();
127     if( res.compareTo(LOGIN)==0) return LOGIN;
        proyecto = manejadorProyecto.datosProyecto(idProyectoElegido);
129     if( proyecto != null){
          ((SessionMap)session).put("idP", proyecto.getIdProyecto());
131       ((SessionMap)session).put("nombreProyecto",proyecto.getNombre()
              );
          return SUCCESS;
133     }else{
          return ERROR;
135     }
      }catch(Exception e){
137     e.printStackTrace();
        return ERROR;
139   }
    }
141

    /**
143  * Obtiene informacion del proyecto con identificador idProyecto.
```

```
     * @return
145   */
    public String datosProyecto(){
147   try{
        proyecto = manejadorProyecto.datosProyecto(this.
            getIdProyectoElegido());
149     idJefe = proyecto.getJefe().getIdUsuario();
        /**
151      * Aqui va a ser el rol de Administrador de Proyectos
         */
153     jefes = manejadorUsuario.usuariosSistemaRol("Rol C");
        proyectosUsuario();
155   }catch(Exception e){
        e.printStackTrace();
157   }
      return SUCCESS;
159 }

161 public String guardarProyecto(){
      try{
163     if( this.getIdProyectoElegido()>0){
          Usuario usr = manejadorUsuario.buscar(idJefe);
165       proyecto.setJefe(usr);
          proyecto = manejadorProyecto.actualizarProyecto(this.getProyecto
              ());
167       this.addActionMessage("Proyecto Actualizado");
        }else{
169       proyectosUsuario();
          return NONE;
171     }
      }catch(Exception e){
173     e.printStackTrace();
        return ERROR;
175   }
      return SUCCESS;
177 }

179 public String eliminarProyecto(){
      try{
181     if(this.getIdProyectoElegido()>0){
          proyecto = manejadorProyecto.eliminarProyecto(this.
              getIdProyectoElegido());
183       proyecto = null;
          proyectosUsuario();
185       this.addActionMessage("Proyecto Eliminado");
        }else{
187       proyectosUsuario();
          return NONE;
189     }

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

```
193       }
          return SUCCESS;
195     }

197     public String listaUsuarios(){
          try{
199         manejadorUsuario = new ManejadorUsuario();
            usuarios = manejadorUsuario.usuariosSistemaRol("Rol C");
201       }catch(Exception e){
            e.printStackTrace();
203       }
          return "success";
205     }

207     /**
         * Llama al manejador de Proyectos para que almacene el nuevo proyecto
209      * en el medio de almacenamiento.
         * @return
211      */
        public String registrarProyecto(){
213       try{
            manejadorUsuario = new ManejadorUsuario();
215         Usuario jefe = manejadorUsuario.buscar(idJefe);
            this.getProyecto().setJefe(jefe);
217         this.addActionMessage("Proyecto Registrado");
          }catch(Exception e){
219         e.printStackTrace();
            return ERROR;
221       }
          return SUCCESS;
223     }


225

        /**
227      * Verifica que el usuario tenga los permisos necesarios
         * para crear un proyecto nuevo
229      * Por el momento no hacemos nada
         */
231     public String verificarPermisos(){
          listaUsuarios();
233       return SUCCESS;
        }
235

        /**
237      * Recupera los detalles del proyecto para
         * mostrarlos en una vista
239      * @return
         */
241     public String detallesProyecto(){
          try{
243         proyecto = manejadorProyecto.datosProyecto(this.
                getIdProyectoElegido());
```

```
      idJefe = proyecto.getJefe().getIdUsuario();
      /**
       * Aqui va a ser el rol de Administrador de Proyectos
       */
      jefes = manejadorUsuario.usuariosSistemaRol("Gestor de Proyecto");
      proyecto = manejadorProyecto.cargarAsociaciones(proyecto);
    }catch(Exception e){
      e.printStackTrace();
      return ERROR;
    }

    return SUCCESS;
  }

  @Override
  public void setSession(Map<String, Object> arg0) {
    // TODO Auto-generated method stub
    session = arg0;
  }

  public Usuario getUser() {
    return user;
  }

  public void setUser(Usuario user) {
    this.user = user;
  }

}
```

src/com/gestion/vista/proyecto/AdministrarProyectoAction.java

```
package com.gestion.vista.rol;

import com.gestion.modelo.manejadorBd.rol.ManejadorRol;
import com.gestion.modelo.rol.Rol;
import com.opensymphony.xwork2.ActionSupport;

public class RolAction extends ActionSupport{
  private Rol rol;
  private String nombreRol;

  private ManejadorRol manejadorRol;

  public RolAction(){
    manejadorRol = new ManejadorRol();
  }

  public Rol getRol() {
    return rol;
  }

  public void setRol(Rol rol) {
```

```
22      this.rol = rol;
    }
24
    public String getNombreRol() {
26      return nombreRol;
    }
28
    public void setNombreRol(String nombreRol) {
30      this.nombreRol = nombreRol;
    }
32
    public String guardarRol(){
34      try{
          rol = manejadorRol.add(rol);
36      }catch(Exception e){
          e.printStackTrace();
38        return ERROR;
        }
40      return SUCCESS;
    }
42
    public String modificarEliminarRol(){
44      try{
          rol = manejadorRol.find(nombreRol);
46      }catch(Exception e){
          e.printStackTrace();
48        return ERROR;
        }
50      return SUCCESS;
    }
52
    public String guardarCambiosRol(){
54      try{
          rol = manejadorRol.actualizarRol(rol);
56      }catch(Exception e){
          e.printStackTrace();
58        return ERROR;
        }
60      return SUCCESS;
    }
62
    public String eliminarRol(){
64      try{
          rol = manejadorRol.eliminarRol(rol);
66      }catch(Exception e){
          e.printStackTrace();
68        return ERROR;
        }
70      return SUCCESS;
    }
72  }
```

src/com/gestion/vista/rol/RolAction.java

```
package com.gestion.vista.rol;

import java.util.List;

import com.gestion.modelo.manejadorBd.rol.ManejadorRol;
import com.gestion.modelo.rol.Rol;
import com.opensymphony.xwork2.ActionSupport;

public class RolSistemaAction extends ActionSupport{
  private List<Rol> roles;
  private String nombreRol;
  private Rol rol;

  private ManejadorRol manejadorRol;

  public RolSistemaAction(){
    manejadorRol = new ManejadorRol();
  }

  public List<Rol> getRoles() {
    return roles;
  }

  public void setRoles(List<Rol> roles) {
    this.roles = roles;
  }

  public String getNombreRol() {
    return nombreRol;
  }

  public void setNombreRol(String nombreRol) {
    this.nombreRol = nombreRol;
  }

  public Rol getRol() {
    return rol;
  }

  public void setRol(Rol rol) {
    this.rol = rol;
  }

  public String rolesSistema(){
    try{
      roles = manejadorRol.lista();
    }catch(Exception e){
      e.printStackTrace();
      return ERROR;
```

```
50      }
        return SUCCESS;
52    }

54    public String detallesRol(){
        try{
56        rol = manejadorRol.find(nombreRol);
        }catch(Exception e){
58        e.printStackTrace();
          return ERROR;
60      }
        return SUCCESS;
62    }
}
```

src/com/gestion/vista/rol/RolSistemaAction.java

```
1  package com.gestion.vista.usuarios;

3  import java.util.Map;

5  import javax.servlet.ServletContext;
   import javax.servlet.http.HttpServletRequest;
7  import javax.servlet.http.HttpServletResponse;

9  import org.apache.struts2.dispatcher.SessionMap;
   import org.apache.struts2.interceptor.ServletRequestAware;
11 import org.apache.struts2.interceptor.ServletResponseAware;
   import org.apache.struts2.interceptor.SessionAware;
13 import org.apache.struts2.util.ServletContextAware;

15 import com.opensymphony.xwork2.ActionSupport;

17 public class LogoutAction extends ActionSupport  implements SessionAware
       , ServletRequestAware,
   ServletResponseAware, ServletContextAware{
19   private Map session;
     @Override
21   public void setSession(Map<String, Object> arg0) {
       this.session = arg0;
23
     }
25   @Override
     public void setServletRequest(HttpServletRequest arg0) {
27     // TODO Auto-generated method stub

29   }
     @Override
31   public void setServletResponse(HttpServletResponse arg0) {
       // TODO Auto-generated method stub
33
     }
35   @Override
```

```
  public void setServletContext(ServletContext arg0) {
37    // TODO Auto-generated method stub

39  }

41  public String salirDelSistema(){
    if ((SessionMap)session instanceof org.apache.struts2.dispatcher.
        SessionMap) {
43        try {
              ((org.apache.struts2.dispatcher.SessionMap) session).
                  invalidate();
45      } catch (IllegalStateException e) {
        e.printStackTrace();
47      }
        System.out.println("Sesion terminada");
49    }
    return SUCCESS;
51  }
}
```

src/com/gestion/vista/usuarios/LogoutAction.java

```
package com.gestion.vista.usuarios;
2
import java.util.List;
4
import com.gestion.modelo.manejadorBd.usuario.ManejadorUsuario;
6 import com.gestion.modelo.usuario.Usuario;

8 public class AdministrarUsuariosAction {
  private List<Usuario> usuariosProyecto;
10  private String idUsuarioElegido;
  private Usuario usuario;
12  private ManejadorUsuario manejadorUsuario;

14  public AdministrarUsuariosAction(){
    manejadorUsuario = new ManejadorUsuario();
16    idUsuarioElegido = "";
  }
18

20  public List<Usuario> getUsuariosProyecto() {
    return usuariosProyecto;
22  }

24

  public void setUsuariosProyecto(List<Usuario> usuariosProyecto) {
26    this.usuariosProyecto = usuariosProyecto;
  }
28

30  public String getIdUsuarioElegido() {
    return idUsuarioElegido;
```

```
32    }
34
      public void setIdUsuarioElegido(String idUsuarioElegido) {
36      this.idUsuarioElegido = idUsuarioElegido;
      }
38
      public Usuario getUsuario() {
40      return usuario;
      }
42
44    public void setUsuario(Usuario usuario) {
        this.usuario = usuario;
46    }

48    /**
       * Obtiene los datos del usuario con el identificador idUsuarioElegido
           .
50     * @return
       */
52    public String datosUsuario(){
        try{
54        usuario = manejadorUsuario.buscar(idUsuarioElegido);
          usuariosProyecto = manejadorUsuario.listaUsuarios(1);
56      }catch(Exception e){
          e.printStackTrace();
58      }
        return "success";
60    }
      /**
62     * Borrar del medio de almacenamiento el usuario con el identificador
       * idUsuarioElegido.
64     * @return
       */
66    public String eliminarUsuario(){
        try{
68        usuario = manejadorUsuario.eliminarUsuario(this.getUsuario().
              getIdUsuario());
        }catch(Exception e){
70        e.printStackTrace();
        }
72      return "success";
      }
74
      /**
76     * Guarda en el medio de almacenamiento a usuario modificado.
       * @return
78     */
      public String guardarUsuario(){
80      try{
          usuario = manejadorUsuario.actualizarUsuario(this.getUsuario());
```

```
82      }catch(Exception e){
          e.printStackTrace();
84      }
        return "success";
86    }
      /**
88     * Consulta los usuarios participantes del
       * proyecto actual.
90     * @return
       */
92    public String usuariosProyecto(){
        try{
94        usuario = null;
          usuariosProyecto = manejadorUsuario.listaUsuarios(1);
96      }catch(Exception e){
          e.printStackTrace();
98      }
        return "success";
100   }

102 }
```

src/com/gestion/vista/usuarios/AdministrarUsuariosAction.java

```
    package com.gestion.vista.usuarios;
2
    import java.util.ArrayList;
4   import java.util.List;

6   import com.gestion.modelo.manejadorBd.rol.ManejadorRol;
    import com.gestion.modelo.manejadorBd.usuario.ManejadorUsuario;
8   import com.gestion.modelo.rol.Rol;
    import com.gestion.modelo.usuario.Accion;
10  import com.gestion.util.PermisoDTO;
    import com.opensymphony.xwork2.Action;
12
    public class SeguridadAction {
14    private ManejadorUsuario manejadorUsuario;
      private String rolActual;
16    private ManejadorRol manejadorRol;
      private List<Rol> roles = new ArrayList<Rol>();
18    private List<PermisoDTO> permisos;
      private String idAccion;
20    private String rolAccion;
      private Accion nuevaAccion;
22
      public SeguridadAction(){
24      manejadorRol = new ManejadorRol();
        manejadorUsuario = new ManejadorUsuario();
26      roles = manejadorRol.lista();
      }
28
      public String verListaRoles(){
```

```
30      roles  = manejadorRol.lista();
        return Action.SUCCESS;
32    }

34    public String verPermisosDeRol(){
        System.out.println("Ver permisos");
36      permisos = manejadorUsuario.permisosDTOXRol(rolActual);
        System.out.println(permisos.size());
38      return Action.SUCCESS;
      }
40
      public String guardarCambiosPermisos(){
42      try{
          System.out.println("Cambiando permisos para: "+this.idAccion+", "+
              this.rolAccion);
44        manejadorUsuario.cambiarPermiso(Integer.valueOf(this.idAccion),
              this.rolAccion);
          permisos = manejadorUsuario.permisosDTOXRol(rolAccion);
46
        }catch (Exception e) {
48        // TODO: handle exception
        }
50      return Action.SUCCESS;
      }
52
      public String registrarAccion(){
54      nuevaAccion.setId(0);
        manejadorUsuario.guardaObjeto(nuevaAccion);
56      permisos = manejadorUsuario.permisosDTOXRol(rolActual);
        return Action.SUCCESS;
58    }

60    public String modificarEliminarAccion(){
        nuevaAccion = manejadorUsuario.getAccion(Integer.valueOf(this.
            idAccion));
62      System.out.println(nuevaAccion.getId());
        this.idAccion = nuevaAccion.getId().toString();
64      return Action.SUCCESS;
      }
66
      public String eliminarAccion(){
68      nuevaAccion.setId(Integer.valueOf(this.idAccion));
        manejadorUsuario.deleteObject(nuevaAccion);
70      System.out.println("Accion eliminada");
        return Action.SUCCESS;
72    }

74    public String actualizarAccion(){
        System.out.println(this.nuevaAccion.getId());
76      nuevaAccion.setId(Integer.valueOf(this.idAccion));
        manejadorUsuario.actualizaObjeto(this.nuevaAccion);
78      return Action.SUCCESS;
```

```
    }

    public String getRolActual() {
      return rolActual;
    }

    public void setRolActual(String rolActual) {
      this.rolActual = rolActual;
    }

    public List<Rol> getRoles() {
      return roles;
    }

    public void setRoles(List<Rol> roles) {
      this.roles = roles;
    }

    public List<PermisoDTO> getPermisos() {
      return permisos;
    }

    public void setPermisos(List<PermisoDTO> permisos) {
      this.permisos = permisos;
    }

    public String getIdAccion() {
      return idAccion;
    }

    public void setIdAccion(String idAccion) {
      this.idAccion = idAccion;
    }

    public String getRolAccion() {
      return rolAccion;
    }

    public void setRolAccion(String rolAccion) {
      this.rolAccion = rolAccion;
    }

    public Accion getNuevaAccion() {
      return nuevaAccion;
    }

    public void setNuevaAccion(Accion nuevaAccion) {
      this.nuevaAccion = nuevaAccion;
    }
}
```

src/com/gestion/vista/usuarios/SeguridadAction.java

```
package com.gestion.vista.usuarios;

import java.util.Map;

import javax.servlet.ServletContext;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

import org.apache.struts2.ServletActionContext;
import org.apache.struts2.dispatcher.SessionMap;
import org.apache.struts2.interceptor.ServletRequestAware;
import org.apache.struts2.interceptor.ServletResponseAware;
import org.apache.struts2.interceptor.SessionAware;
import org.apache.struts2.util.ServletContextAware;

import com.gestion.modelo.manejadorBd.usuario.ManejadorUsuario;
import com.gestion.modelo.usuario.Usuario;
import com.opensymphony.xwork2.ActionContext;
import com.opensymphony.xwork2.ActionSupport;

public class ValidarUsuarioAction extends ActionSupport  implements
    SessionAware, ServletRequestAware,
  ServletResponseAware, ServletContextAware{
  /**
   *
   */
  private static final long serialVersionUID = 1L;
  /**
   *
   */
  private String login;
  private String password;
  private Map session;

  private ManejadorUsuario manejadorUsuario;


  public ValidarUsuarioAction(){
    manejadorUsuario = new ManejadorUsuario();
  }

  public String getLogin() {
    return login;
  }

  public void setLogin(String login) {
    this.login = login;
  }

  public String getPassword() {
```

```
    return password;
  }

  public void setPassword(String password) {
    this.password = password;
  }

  public String login(){
    return INPUT;
  }

  public String inicio(){
    return SUCCESS;
  }


  @Override
  public void setSession(Map<String, Object> arg0) {
    // TODO Auto-generated method stub
    this.session = arg0;
  }

  @Override
  public void setServletRequest(HttpServletRequest arg0) {
    // TODO Auto-generated method stub

  }

  @Override
  public void setServletResponse(HttpServletResponse arg0) {
    // TODO Auto-generated method stub

  }

  @Override
  public void setServletContext(ServletContext arg0) {
    // TODO Auto-generated method stub

  }
  public String validarUsuario(){
    try{
      Usuario user = manejadorUsuario.usuario(login);
      user = manejadorUsuario.cargarRoles(user);
      /**
       * si user es null ya no se compara password, no tiene caso
       */
      if( (user != null) && (user.getPass_word().compareTo(password)==0)
          ){
        if(((SessionMap)session).get("usuario")==null){
          System.out.println("No esta en session");
          ((SessionMap)session).put("usuario", user);
```

```
            ((SessionMap)session).put("nombreUsuario", user.
                getNombreCompleto());
102         System.out.println("Ahora esta en session");
          }
104     }else{
          this.addActionMessage("Login o password incorrectos");
106       return LOGIN;
          }
108   }catch(Exception e){
        e.printStackTrace();
110     password="";
        return ERROR;
112   }
      password="";
114   return SUCCESS;
    }
116

118 }
```

src/com/gestion/vista/usuarios/ValidarUsuarioAction.java

```
<!DOCTYPE validators PUBLIC
2     "-//OpenSymphony Group//XWork Validator 1.0.2//EN"
      "http://www.opensymphony.com/xwork/xwork-validator-1.0.2.dtd">
4 <validators>
      <field name="login">
6         <field-validator type="requiredstring">
          <param name="trim">true</param>
8             <message>Se necesita un login</message>
          </field-validator>
10    </field>
      <field name="password">
12        <field-validator type="requiredstring">
              <message>Introduzca un password</message>
14        </field-validator>
      </field>
16 </validators>
```

src/com/gestion/vista/usuarios/ValidarUsuarioAction-validation.xml

```
package com.gestion.vista.usuarios;
2
public class PruebaVariosBotonesAction {
4   private String mensaje;

6   public String getMensaje() {
      return mensaje;
8   }

10  public void setMensaje(String mensaje) {
      this.mensaje = mensaje;
12  }
```

```
  /**
   * Accion que se ejecutara al llamara al boton uno
   * @return
   */
  public String accion1(){
    mensaje = "Se ejecuto la accion uno";
    return "success";
  }
  /**
   * Accion que se ejecutara al llamara al boton dos
   * @return
   */
  public String accion2(){
    mensaje = "Se ejecuto la accion dos";
    return "success";
  }
  /**
   * Accion que se ejecutara al llamara al boton tres
   * @return
   */
  public String accion3(){
    mensaje = "Se ejecuto la accion tres";
    return "success";
  }
}
```

src/com/gestion/vista/usuarios/PruebaVariosBotonesAction.java

```
package com.gestion.vista.actividad;

import java.util.List;

import com.gestion.modelo.manejadorBd.fase.ManejadorActividad;
import com.gestion.modelo.manejadorBd.fase.ManejadorDisciplina;
import com.gestion.modelo.proyecto.Actividad;
import com.gestion.modelo.proyecto.Disciplina;
import com.opensymphony.xwork2.ActionSupport;

public class ActividadAction extends ActionSupport{
  private Actividad actividad;
  private List<Actividad> actividades;
  private List<Disciplina> disciplinas;

  private long idActividad;
  private long idDisciplina;

  private ManejadorActividad manejadorActividad;
  private ManejadorDisciplina manejadorDisciplina;

  public ActividadAction(){
    manejadorActividad = new ManejadorActividad();
    manejadorDisciplina = new ManejadorDisciplina();
  }
```

```
27    public Actividad getActividad() {
        return actividad;
29    }

31    public void setActividad(Actividad actividad) {
        this.actividad = actividad;
33    }

35    public List<Actividad> getActividades() {
        return actividades;
37    }

39    public void setActividades(List<Actividad> actividades) {
        this.actividades = actividades;
41    }

43    public long getIdActividad() {
        return idActividad;
45    }

47    public void setIdActividad(long idActividad) {
        this.idActividad = idActividad;
49    }

51    public List<Disciplina> getDisciplinas() {
        return disciplinas;
53    }

55    public void setDisciplinas(List<Disciplina> disciplinas) {
        this.disciplinas = disciplinas;
57    }

59    public long getIdDisciplina() {
        return idDisciplina;
61    }

63    public void setIdDisciplina(long idDisciplina) {
        this.idDisciplina = idDisciplina;
65    }

67    public String actividadesRup(){
        try{
69        actividades = manejadorActividad.lista();
        }catch(Exception e){
71        e.printStackTrace();
          return ERROR;
73      }
        return SUCCESS;
75    }

77    public String detallesActividad(){
```

```
    try{
79    actividad = manejadorActividad.buscarActividad(idActividad);
      actividad = manejadorActividad.cargarArtefactos(actividad);
81  }catch(Exception e){
      e.printStackTrace();
83    return ERROR;
    }
85  return SUCCESS;
  }
87}
```

src/com/gestion/vista/actividad/ActividadAction.java

```
 1package com.gestion.vista.actividad;

 3import java.util.HashSet;
  import java.util.Iterator;
 5import java.util.List;
  import java.util.Set;
 7
  import com.gestion.modelo.manejadorBd.artefacto.ManejadorArtefacto;
 9import com.gestion.modelo.manejadorBd.fase.ManejadorActividad;
  import com.gestion.modelo.manejadorBd.fase.ManejadorDisciplina;
11import com.gestion.modelo.manejadorBd.rol.ManejadorRol;
  import com.gestion.modelo.proyecto.Actividad;
13import com.gestion.modelo.proyecto.Artefacto;
  import com.gestion.modelo.proyecto.Disciplina;
15import com.gestion.modelo.rol.Rol;
  import com.opensymphony.xwork2.ActionSupport;
17
  public class ModificarEliminarActividadAction extends ActionSupport{
19  private Actividad actividad;
    private List<Disciplina> disciplinas;
21  private long idDisciplina;
    private long idActividad;
23  private String nombreRol;

25  private List<Rol> roles;
    private List<String> artefactosEntrada;
27  private List<String> artefactosProducidos;
    private List<Artefacto> artefactos;
29
    private ManejadorActividad manejadorActividad;
31  private ManejadorDisciplina manejadorDisciplina;
    private ManejadorArtefacto manejadorArtefacto;
33  private ManejadorRol manejadorRol;

35  public ModificarEliminarActividadAction(){
      manejadorActividad = new ManejadorActividad();
37    manejadorArtefacto = new ManejadorArtefacto();
      manejadorDisciplina = new ManejadorDisciplina();
39    manejadorRol = new ManejadorRol();
    }
```

```
41
    public Actividad getActividad() {
43    return actividad;
    }
45
    public void setActividad(Actividad actividad) {
47    this.actividad = actividad;
    }
49
    public List<Disciplina> getDisciplinas() {
51    return disciplinas;
    }
53
    public void setDisciplinas(List<Disciplina> disciplinas) {
55    this.disciplinas = disciplinas;
    }
57
    public long getIdDisciplina() {
59    return idDisciplina;
    }
61
    public void setIdDisciplina(long idDisciplina) {
63    this.idDisciplina = idDisciplina;
    }
65
    public long getIdActividad() {
67    return idActividad;
    }
69
    public void setIdActividad(long idActividad) {
71    this.idActividad = idActividad;
    }
73
    public List<String> getArtefactosEntrada() {
75    return artefactosEntrada;
    }
77
    public void setArtefactosEntrada(List<String> artefactosEntrada) {
79    this.artefactosEntrada = artefactosEntrada;
    }
81
    public List<String> getArtefactosProducidos() {
83    return artefactosProducidos;
    }
85
    public void setArtefactosProducidos(List<String> artefactosProducidos)
        {
87    this.artefactosProducidos = artefactosProducidos;
    }
89
    public List<Artefacto> getArtefactos() {
91    return artefactos;
```

```
  }
93
  public void setArtefactos(List<Artefacto> artefactos) {
95  this.artefactos = artefactos;
  }
97
  public String getNombreRol() {
99  return nombreRol;
  }
101
  public void setNombreRol(String nombreRol) {
103  this.nombreRol = nombreRol;
  }
105
  public List<Rol> getRoles() {
107  return roles;
  }
109
  public void setRoles(List<Rol> roles) {
111  this.roles = roles;
  }
113
  public String modificarEliminarActividad(){
115  try{
       actividad = manejadorActividad.buscarActividad(idActividad);
117    disciplinas = manejadorDisciplina.lista();
       actividad = manejadorActividad.cargarArtefactos(actividad);
119    artefactos = manejadorArtefacto.lista();
       roles = manejadorRol.lista();
121  }catch(Exception e){
       e.printStackTrace();
123    return ERROR;
     }
125  return SUCCESS;
  }
127
  public String guardarActividadModificada(){
129  try{
       Set<Artefacto> requeridos = new HashSet<Artefacto>();
131    Set<Artefacto> producidos = new HashSet<Artefacto>();
       Disciplina disciplina = manejadorDisciplina.buscar(idDisciplina);
133    Rol rol = manejadorRol.find(nombreRol);

135

       Artefacto a = null;
137    Iterator<String> it = artefactosEntrada.iterator();
       while(it.hasNext()){
139      a = manejadorArtefacto.find(it.next());
         if( a !=  null) requeridos.add(a);
141    }
       it = artefactosProducidos.iterator();
143    while(it.hasNext()){
```

```
        a = manejadorArtefacto.find(it.next());
145     if(a != null) producidos.add(a);
      }
147   actividad.setDisciplina(disciplina);
      actividad.setRequeridos(requeridos);
149   actividad.setProducidos(producidos);
      actividad.setRol(rol);
151   actividad = manejadorActividad.actualizarActividad(actividad);
    }catch(Exception e){
153   e.printStackTrace();
      return ERROR;
155 }
    return SUCCESS;
157 }

159 }
```

src/com/gestion/vista/actividad/ModificarEliminarActividadAction.java

```
  package com.gestion.vista.actividad;
2
  import java.util.ArrayList;
4 import java.util.HashSet;
  import java.util.Iterator;
6 import java.util.List;
  import java.util.Set;
8
  import com.gestion.modelo.manejadorBd.artefacto.ManejadorArtefacto;
10 import com.gestion.modelo.manejadorBd.fase.ManejadorActividad;
  import com.gestion.modelo.manejadorBd.fase.ManejadorDisciplina;
12 import com.gestion.modelo.manejadorBd.rol.ManejadorRol;
  import com.gestion.modelo.proyecto.Actividad;
14 import com.gestion.modelo.proyecto.Artefacto;
  import com.gestion.modelo.proyecto.Disciplina;
16 import com.gestion.modelo.rol.Rol;
  import com.opensymphony.xwork2.ActionSupport;
18
  public class AgregarActividadAction extends ActionSupport{
20   private List<Disciplina> disciplinas;
     private long idDisciplina;
22   private List<Artefacto> artefactos;
     private List<String> artefactosEntrada;
24   private List<String> artefactosProducidos;
     private Actividad actividad;
26   private List<Rol> roles;
     private String nombreRol;
28


30   private ManejadorDisciplina manejadorDisciplina;
     private ManejadorArtefacto manejadorArtefacto;
32   private ManejadorActividad manejadorActividad;
     private ManejadorRol manejadorRol;
34
```

```
  public AgregarActividadAction(){
36    manejadorActividad = new ManejadorActividad();
      manejadorArtefacto = new ManejadorArtefacto();
38    manejadorDisciplina = new ManejadorDisciplina();
      manejadorRol = new ManejadorRol();
40  }

42

  public List<Disciplina> getDisciplinas() {
44    return disciplinas;
  }
46

48  public void setDisciplinas(List<Disciplina> disciplinas) {
      this.disciplinas = disciplinas;
50  }

52

  public long getIdDisciplina() {
54    return idDisciplina;
  }
56

58  public void setIdDisciplina(long idDisciplina) {
      this.idDisciplina = idDisciplina;
60  }

62

  public List<Artefacto> getArtefactos() {
64    return artefactos;
  }
66

68  public void setArtefactos(List<Artefacto> artefactos) {
      this.artefactos = artefactos;
70  }

72

  public List<String> getArtefactosEntrada() {
74    return artefactosEntrada;
  }
76

78  public void setArtefactosEntrada(List<String> artefactosEntrada) {
      this.artefactosEntrada = artefactosEntrada;
80  }

82

  public List<String> getArtefactosProducidos() {
84    return artefactosProducidos;
  }
86
```

```
88    public void setArtefactosProducidos(List<String> artefactosProducidos)
          {
        this.artefactosProducidos = artefactosProducidos;
90    }

92    public Actividad getActividad() {
        return actividad;
94    }

96
      public void setActividad(Actividad actividad) {
98      this.actividad = actividad;
      }
100
      public List<Rol> getRoles() {
102     return roles;
      }
104

106   public void setRoles(List<Rol> roles) {
        this.roles = roles;
108   }

110
      public String getNombreRol() {
112     return nombreRol;
      }
114

116   public void setNombreRol(String nombreRol) {
        this.nombreRol = nombreRol;
118   }

120
      public String nuevaActividad(){
122     try{
          disciplinas = manejadorDisciplina.lista();
124       artefactos = manejadorArtefacto.lista();
          roles = manejadorRol.lista();
126     }catch(Exception e){
          e.printStackTrace();
128       return ERROR;
        }
130     return SUCCESS;
      }
132   public String agregarActividad(){
        try{
134       Disciplina elegida = manejadorDisciplina.buscar(idDisciplina);
          System.out.println(elegida.getNombre());
136       System.out.println(elegida.getDescripcion());
          actividad.setDisciplina(elegida);
```

```
138
        Rol rol = manejadorRol.find(nombreRol);
140     System.out.println("Rol: "+rol.getNombreRol());
        actividad.setRol(rol);
142
        actividad = manejadorActividad.agregarActividad(actividad);
144
        Iterator<String> it = artefactosEntrada.iterator();
146     Set<Artefacto> artefactosRequeridos = new HashSet<Artefacto>();
        Artefacto a = null;
148     String nombreArtefacto = "";
        while(it.hasNext()){
150       nombreArtefacto = it.next();
          System.out.println(nombreArtefacto);
152       a = manejadorArtefacto.find(nombreArtefacto);
          if(a != null){
154         artefactosRequeridos.add(a);
          }
156     }
        actividad.setRequeridos(artefactosRequeridos);
158
        Set<Artefacto> artefactosSalida = new HashSet<Artefacto>();
160
        it = artefactosProducidos.iterator();
162
        while(it.hasNext()){
164       nombreArtefacto = it.next();
          a = manejadorArtefacto.find(nombreArtefacto);
166       if( a != null){
            artefactosSalida.add(a);
168       }
        }
170     actividad.setProducidos(artefactosSalida);

172     System.out.println("Artefactos de entrada: "+artefactosRequeridos.
            toString());
        System.out.println("Artefactos de salida: "+ artefactosSalida.
            toString());
174     actividad = manejadorActividad.actualizarActividad(actividad);
        System.out.println("Agregada");
176   }catch(Exception e){
        e.printStackTrace();
178     return ERROR;
      }
180   return SUCCESS;
    }
182 }
```

src/com/gestion/vista/actividad/AgregarActividadAction.java

```
  package com.gestion.vista.artefactos;
2
  import java.util.HashSet;
```

```
4 import java.util.List;
  import java.util.Map;
6
  import org.apache.struts2.dispatcher.SessionMap;
8 import org.apache.struts2.interceptor.SessionAware;

10 import com.gestion.modelo.manejadorBd.artefacto.ManejadorArtefacto;
  import com.gestion.modelo.manejadorBd.rol.ManejadorRol;
12 import com.gestion.modelo.proyecto.Artefacto;
  import com.gestion.modelo.rol.Rol;
14 import com.gestion.modelo.usuario.Usuario;
  import com.opensymphony.xwork2.ActionSupport;
16
  public class ArtefactosSistemaAction extends ActionSupport implements
     SessionAware{
18   private List<Artefacto> artefactos;
     ManejadorArtefacto manejadorArtefacto;
20   ManejadorRol manejadorRol;
     private String nombreArtefacto;
22   private String nombreRol;
     private Rol rol;
24   private Artefacto artefacto;

26   Map session;

28   public ArtefactosSistemaAction(){
       manejadorArtefacto = new ManejadorArtefacto();
30     manejadorRol = new ManejadorRol();
     }
32
     public List<Artefacto> getArtefactos() {
34     return artefactos;
     }
36
     public void setArtefactos(List<Artefacto> artefactos) {
38     this.artefactos = artefactos;
     }
40
     public String getNombreArtefacto() {
42     return nombreArtefacto;
     }
44
     public void setNombreArtefacto(String nombreArtefacto) {
46     this.nombreArtefacto = nombreArtefacto;
     }
48
     public Artefacto getArtefacto() {
50     return artefacto;
     }
52
     public void setArtefacto(Artefacto artefacto) {
54     this.artefacto = artefacto;
```

```
  }

  public String getNombreRol() {
    return nombreRol;
  }

  public void setNombreRol(String nombreRol) {
    this.nombreRol = nombreRol;
  }

  public Rol getRol() {
    return rol;
  }

  public void setRol(Rol rol) {
    this.rol = rol;
  }

  public String artefactosSistema(){
    try{
      System.out.println("En artefactosSistema()");
      artefactos = manejadorArtefacto.lista();
    }catch(Exception e){
      e.printStackTrace();
      return ERROR;
    }
    return SUCCESS;
  }

  public String artefactosProyecto(){
    try{
      if((session instanceof SessionMap) && (((SessionMap)session).get("
          idP")!=null)){
        long idProyecto = (Long)(((SessionMap)session).get("idP"));
      }
    }catch(Exception e){
      e.printStackTrace();
      return ERROR;
    }
    return SUCCESS;
  }

  public String detallesArtefacto(){
    try{
      artefacto = manejadorArtefacto.find(nombreArtefacto);
    }catch(Exception e){
      e.printStackTrace();
      return ERROR;
    }
    return SUCCESS;
  }
```

```
106   public String detallesRol(){
        try{
108       rol = manejadorRol.find(nombreRol);
        }catch(Exception e){
110       e.printStackTrace();
          return ERROR;
112     }
        return SUCCESS;
114   }

116   @Override
      public void setSession(Map<String, Object> arg0) {
118     // TODO Auto-generated method stub
        this.session = arg0;
120   }
}
```

src/com/gestion/vista/artefactos/ArtefactosSistemaAction.java

```
1  package com.gestion.vista.artefactos;


3
   import java.io.File;
5  import java.io.FileInputStream;
   import java.sql.Blob;
7  import java.sql.Date;
   import java.util.List;
9
   import javax.servlet.ServletContext;
11
   import org.apache.struts2.ServletActionContext;
13 import org.hibernate.Hibernate;


15
   import com.gestion.modelo.manejadorBd.artefacto.ManejadorArtefacto;
17 import com.gestion.modelo.manejadorBd.rol.ManejadorRol;
   import com.gestion.modelo.proyecto.Artefacto;
19 import com.gestion.modelo.rol.Rol;
   import com.opensymphony.xwork2.ActionSupport;
21
   public class AgregarArtefactoAction extends ActionSupport{
23   private Artefacto artefacto;
     private List<Rol> roles;
25   private String rolElegido;
     private File attachment;
27   private String attachmentFileName;
     private String attachmentContentType;
29



31
     private ManejadorArtefacto manejadorArtefacto;
33   private ManejadorRol manejadorRol;
```

```
35  public AgregarArtefactoAction(){
      manejadorArtefacto = new ManejadorArtefacto();
37    manejadorRol = new ManejadorRol();
    }
39
    public Artefacto getArtefacto() {
41    return artefacto;
    }
43  public void setArtefacto(Artefacto artefacto) {
      this.artefacto = artefacto;
45  }
    public List<Rol> getRoles() {
47    return roles;
    }
49  public void setRoles(List<Rol> roles) {
      this.roles = roles;
51  }
    public String getRolElegido() {
53    return rolElegido;
    }
55  public void setRolElegido(String rolElegido) {
      this.rolElegido = rolElegido;
57  }

59  public File getAttachment() {
      return attachment;
61  }

63  public void setAttachment(File attachment) {
      this.attachment = attachment;
65  }

67  public String getAttachmentFileName() {
      return attachmentFileName;
69  }

71  public void setAttachmentFileName(String attachmentFileName) {
      this.attachmentFileName = attachmentFileName;
73  }

75  public String getAttachmentContentType() {
      return attachmentContentType;
77  }

79  public void setAttachmentContentType(String attachmentContentType) {
      this.attachmentContentType = attachmentContentType;
81  }

83  public String nuevoArtefacto(){
      try{
85      roles = manejadorRol.lista();
      }catch(Exception e){
```

```
87          e.printStackTrace();
            return ERROR;
89        }
          return SUCCESS;
91      }

93      public String guardarArtefacto(){
          try{
95            Rol a = manejadorRol.find(rolElegido);
              artefacto.setRol(a);
97            artefacto = manejadorArtefacto.add(artefacto);
        }catch(Exception e){
99        e.printStackTrace();
          return ERROR;
101     }
        return SUCCESS;
103   }
    }
```

src/com/gestion/vista/artefactos/AgregarArtefactoAction.java

```
    package com.gestion.vista.artefactos;
2
    import java.util.ArrayList;
4   import java.util.List;

6   import com.gestion.modelo.manejadorBd.artefacto.ManejadorArtefacto;
    import com.gestion.modelo.proyecto.Artefacto;
8   import com.opensymphony.xwork2.ActionSupport;

10  public class SubirDescargarPlantilla extends ActionSupport{
      private List<Artefacto> artefactos;
12    private String nombreArtefacto;

14    private ManejadorArtefacto manejadorArtefacto;
      public SubirDescargarPlantilla(){
16      manejadorArtefacto = new ManejadorArtefacto();
      }
18



20
    }
```

src/com/gestion/vista/artefactos/SubirDescargarPlantilla.java

```
1   package com.gestion.vista.artefactos;

3   import java.util.List;
    import java.util.Map;
5
    import org.apache.struts2.interceptor.SessionAware;
7
    import com.gestion.modelo.manejadorBd.artefacto.ManejadorArtefacto;
```

```
9  import com.gestion.modelo.manejadorBd.rol.ManejadorRol;
   import com.gestion.modelo.proyecto.Artefacto;
11 import com.gestion.modelo.rol.Rol;
   import com.opensymphony.xwork2.ActionSupport;
13
   public class ModificarEliminarArtefactoAction  extends ActionSupport{
15   private Artefacto artefacto;
     private String nombreArtefacto;
17   private List<Rol> roles;
     private String nombreRol;
19
     private ManejadorRol manejadorRol;
21   private ManejadorArtefacto manejadorArtefacto;

23   public ModificarEliminarArtefactoAction(){
       manejadorArtefacto = new ManejadorArtefacto();
25     manejadorRol = new ManejadorRol();
     }
27
     public Artefacto getArtefacto() {
29     return artefacto;
     }
31
     public void setArtefacto(Artefacto artefacto) {
33     this.artefacto = artefacto;
     }
35
     public String getNombreArtefacto() {
37     return nombreArtefacto;
     }
39
     public void setNombreArtefacto(String nombreArtefacto) {
41     this.nombreArtefacto = nombreArtefacto;
     }
43
     public List<Rol> getRoles() {
45     return roles;
     }
47
     public void setRoles(List<Rol> roles) {
49     this.roles = roles;
     }
51
     public String getNombreRol() {
53     return nombreRol;
     }
55
     public void setNombreRol(String nombreRol) {
57     this.nombreRol = nombreRol;
     }
59
     public String modificarEliminarArtefacto(){
```

```
61      try{
          roles = manejadorRol.lista();
63        artefacto = manejadorArtefacto.find(nombreArtefacto);
        }catch(Exception e){
65        e.printStackTrace();
          return ERROR;
67      }
        return SUCCESS;
69    }

71    public String guardarCambiosArtefacto(){
        try{
73        Rol rol = manejadorRol.find(nombreRol);
          if( rol != null){
75          artefacto.setRol(rol);
            artefacto = manejadorArtefacto.actualizarArtefacto(artefacto);
77        }else{
            return ERROR;
79        }
        }catch(Exception e){
81        e.printStackTrace();
          return ERROR;
83      }
        return SUCCESS;
85    }

87    public String eliminarArtefacto(){
        try{
89        manejadorArtefacto.eliminarArtefacto(artefacto);
        }catch(Exception e){
91        e.printStackTrace();
          return ERROR;
93      }
        return SUCCESS;
95    }
  }
```

src/com/gestion/vista/artefactos/ModificarEliminarArtefactoAction.java

```
  package com.gestion.vista.personal;
2
  import java.util.HashSet;
4 import java.util.Iterator;
  import java.util.List;
6 import java.util.Map;
  import java.util.Set;
8
  import org.apache.struts2.dispatcher.SessionMap;
10 import org.apache.struts2.interceptor.SessionAware;

12 import com.gestion.modelo.manejadorBd.proyecto.ManejadorProyecto;
  import com.gestion.modelo.manejadorBd.rol.ManejadorRol;
14 import com.gestion.modelo.manejadorBd.usuario.ManejadorUsuario;
```

```
import com.gestion.modelo.proyecto.Proyecto;
import com.gestion.modelo.rol.Rol;
import com.gestion.modelo.usuario.Usuario;
import com.opensymphony.xwork2.ActionSupport;

public class ModificarEliminarUsuarioAction extends ActionSupport
    implements SessionAware{

  private String idUsuario;
  private Usuario usuario;
  private Proyecto proyecto;
  private List<String> rolesAsignados;
  private List<Rol> rolesSistema;

  private Map session;
  private ManejadorProyecto manejadorProyecto;
  private ManejadorUsuario manejadorUsuario;
  private ManejadorRol manejadorRol;

  public ModificarEliminarUsuarioAction(){
    manejadorProyecto = new ManejadorProyecto();
    manejadorRol = new ManejadorRol();
    manejadorUsuario = new ManejadorUsuario();
  }


  public String getIdUsuario() {
    return idUsuario;
  }


  public void setIdUsuario(String idUsuario) {
    this.idUsuario = idUsuario;
  }


  public Usuario getUsuario() {
    return usuario;
  }


  public void setUsuario(Usuario usuario) {
    this.usuario = usuario;
  }


  public Proyecto getProyecto() {
    return proyecto;
  }


  public void setProyecto(Proyecto proyecto) {
```

```
66      this.proyecto = proyecto;
    }
68

70    public List<String> getRolesAsignados() {
      return rolesAsignados;
72    }

74
    public void setRolesAsignados(List<String> rolesAsignados) {
76      this.rolesAsignados = rolesAsignados;
    }
78

80    public List<Rol> getRolesSistema() {
      return rolesSistema;
82    }

84
    public void setRolesSistema(List<Rol> rolesSistema) {
86      this.rolesSistema = rolesSistema;
    }
88

90    @Override
    public void setSession(Map<String, Object> arg0) {
92      // TODO Auto-generated method stub
      this.session = arg0;
94    }

96    @org.apache.struts2.interceptor.validation.SkipValidation
    public String modificarEliminarUsuario(){
98      try{
        if((session instanceof SessionMap) && (((SessionMap)session).get("
            idP")!=null)){
100       long idProyecto = (Long)(((SessionMap)session).get("idP"));
          proyecto = manejadorProyecto.datosProyecto(idProyecto);
102       proyecto = manejadorProyecto.cargarUsuarios(proyecto);
          rolesSistema = manejadorRol.lista();
104       usuario = new Usuario();
          usuario.setIdUsuario("");
106       usuario.setRoles(new HashSet<Rol>());
        }
108     }catch(Exception e){
        e.printStackTrace();
110     return ERROR;
      }
112     return SUCCESS;
    }
114

    @org.apache.struts2.interceptor.validation.SkipValidation
116   public String datosUsuario(){
```

```
    try{
118   modificarEliminarUsuario();
      usuario = manejadorUsuario.buscar(idUsuario);
120   usuario = manejadorUsuario.cargarRoles(usuario);
      /*
122    * solo para que no se envie
       */
124   usuario.setPass_word("");
    }catch(Exception e){
126   e.printStackTrace();
      return ERROR;
128 }
    return SUCCESS;
130 }

132 public String actualizarUsuario(){
    try{
134   System.out.println("En actualizarUsuario");
      Usuario anterior = manejadorUsuario.buscar(usuario.getIdUsuario())
          ;
136   Iterator<String> it = rolesAsignados.iterator();
      Set<Rol> roles = new HashSet<Rol>();
138   Rol a;

140   while(it.hasNext()){
        a = manejadorRol.find(it.next());
142     if( a != null){
          if(roles.add(a)){
144         System.out.println(a.getNombreRol());
          }
146     }
      }
148
      usuario.setPass_word(anterior.getPass_word());
150   usuario.setRoles(roles);
      usuario = manejadorUsuario.actualizarUsuario(usuario);
152   /*
       * solo para que no se envie el password
154    */
      usuario.setPass_word("");
156


158 }catch(Exception e){
      e.printStackTrace();
160   return ERROR;
    }
162 return SUCCESS;
  }
164
  public String eliminarParticipante(){
166 try{
```

```
        Usuario u = manejadorUsuario.eliminarUsuario(usuario.getIdUsuario
            ());
168     if(u != null){
          u = manejadorUsuario.cargarProyectos(u);
170       if(u.getProyectos().size()>1){
            this.addActionMessage("No de puede eliminar este usuario, " +
172             "porque esta participando en otros proyectos.");
          }else{
174         u = manejadorUsuario.eliminarUsuario(u.getIdUsuario());
          }
176     }
      }catch(Exception e){
178     e.printStackTrace();
        return ERROR;
180   }
      return SUCCESS;
182 }
}
```

src/com/gestion/vista/personal/ModificarEliminarUsuarioAction.java

```
1  <!DOCTYPE validators PUBLIC
       "-//OpenSymphony Group//XWork Validator 1.0.2//EN"
3      "http://www.opensymphony.com/xwork/xwork-validator-1.0.2.dtd">
   <validators>
5      <field name="usuario.idUsuario">
           <field-validator type="requiredstring">
7          <param name="trim">true</param>
               <message>Se necesita el dato CURP</message>
9          </field-validator>
       </field>
11     <field name="usuario.nombre">
           <field-validator type="requiredstring">
13         <param name="trim">true</param>
               <message>Se necesita al menos el nombre</message>
15         </field-validator>
       </field>
17
       <field name="usuario.correo">
19         <field-validator type="requiredstring" short-circuit="true">
             <param name="trim">true</param>
21             <message>Se necesita un correo</message>
           </field-validator>
23         <field-validator type="email">
               <message>Correo no valido</message>
25         </field-validator>
       </field>
27
       <field name="password1">
29         <field-validator type="requiredstring">
               <message>Introduzca un password</message>
31         </field-validator>
       </field>
```

```
33
    <field name="usuario.pass_word">
35      <field-validator type="requiredstring" short-circuit="true">
            <message>Repita el password</message>
37      </field-validator>
        <field-validator type="fieldexpression">
39        <param name="expression">
          password1.equals(usuario.pass_word)
41        </param>
          <message>
43        Los passwords no son iguales
          </message>
45      </field-validator>
    </field>
47 </validators>
```

src/com/gestion/vista/personal/NuevoPersonalAction–validation.xml

```
1  package com.gestion.vista.personal;

3  import java.util.ArrayList;
   import java.util.HashSet;
5  import java.util.Iterator;
   import java.util.List;
7  import java.util.Map;
   import java.util.Set;
9
   import org.apache.struts2.dispatcher.SessionMap;
11 import org.apache.struts2.interceptor.SessionAware;

13 import com.gestion.modelo.manejadorBd.proyecto.ManejadorProyecto;
   import com.gestion.modelo.manejadorBd.rol.ManejadorRol;
15 import com.gestion.modelo.manejadorBd.usuario.ManejadorUsuario;
   import com.gestion.modelo.proyecto.Proyecto;
17 import com.gestion.modelo.rol.Rol;
   import com.gestion.modelo.usuario.Usuario;
19 import com.opensymphony.xwork2.ActionSupport;

21 public class AgregarPersonalProyectoAction extends ActionSupport
      implements SessionAware{
     private Proyecto proyecto;
23   private Usuario usuario;
     private Set<String> usuariosElegidos=new HashSet<String>();
25   private List<Usuario> usuariosSistema = new ArrayList<Usuario>();
     private Set<String> rolesAsignados = new HashSet<String>();
27   private List<Rol> roles = new ArrayList<Rol>();
     private String password1;
29
     private Map session;
31   private ManejadorProyecto manejadorProyecto;
     private ManejadorUsuario manejadorUsuario;
33   private ManejadorRol manejadorRol;
```

```
35    public AgregarPersonalProyectoAction(){
        manejadorProyecto = new ManejadorProyecto();
37      manejadorUsuario = new ManejadorUsuario();
        manejadorRol =  new ManejadorRol();
39    }

41    public Proyecto getProyecto() {
        return proyecto;
43    }

45    public void setProyecto(Proyecto proyecto) {
        this.proyecto = proyecto;
47    }

49    public Set<String> getUsuariosElegidos() {
        return usuariosElegidos;
51    }

53    public void setUsuariosElegidos(Set<String> usuariosElegidos) {
        this.usuariosElegidos = usuariosElegidos;
55    }

57    public Usuario getUsuario() {
        return usuario;
59    }

61    public void setUsuario(Usuario usuario) {
        this.usuario = usuario;
63    }

65    public List<Usuario> getUsuariosSistema() {
        return usuariosSistema;
67    }

69    public void setUsuariosSistema(List<Usuario> usuariosSistema) {
        this.usuariosSistema = usuariosSistema;
71    }

73    public Set<String> getRolesAsignados() {
        return rolesAsignados;
75    }

77    public void setRolesAsignados(Set<String> rolesAsignados) {
        this.rolesAsignados = rolesAsignados;
79    }
      public List<Rol> getRoles() {
81      return roles;
      }
83
      public void setRoles(List<Rol> roles) {
85      this.roles = roles;
      }
```

```
87
   public String getPassword1() {
89   return password1;
   }
91
   public void setPassword1(String password1) {
93   this.password1 = password1;
   }
95
   @Override
97 public void setSession(Map<String, Object> arg0) {
     // TODO Auto-generated method stub
99   this.session = arg0;
   }
101 /**
     * Inicializa la lista de usuarios del sistema
103  * e inicializa el objeto proyecto
     * @return
105  */
   public String participantesProyecto(){
107  try{
       if( session instanceof SessionMap && ((SessionMap)session).get("
           idP") != null){
109      long idProyecto = (Long)((SessionMap)session).get("idP");
         proyecto = manejadorProyecto.datosProyecto(idProyecto);
111      proyecto = manejadorProyecto.cargarUsuarios(proyecto);
         usuariosSistema = manejadorUsuario.lista();
113      usuariosSistema = manejadorUsuario.EmpleadoDisponible(proyecto.
             getIdProyecto());
         roles = manejadorRol.lista();
115    }else{
         return LOGIN;
117    }
     }catch(Exception e){
119    e.printStackTrace();
       return ERROR;
121  }
     return SUCCESS;
123 }

125 public String asignarPersonalProyecto(){
     try{
127    Iterator<String> it = this.getUsuariosElegidos().iterator();

129    if( session instanceof SessionMap && ((SessionMap)session).get("
           idP") != null){
       long idProyecto = (Long)((SessionMap)session).get("idP");
131    proyecto = manejadorProyecto.datosProyecto(idProyecto);

133    Set<Usuario> personalAsignado = new HashSet<Usuario>();
       Usuario usr;
135    Usuario u = (Usuario)((SessionMap)session).get("usuario");
```

```
137           while(it.hasNext()){
                usr = manejadorUsuario.buscar(it.next());
139             if(usr != null && !usr.equals(u)){
                  personalAsignado.add(usr);
141             }
              }
143           proyecto.setListaUsuarios(personalAsignado);
              proyecto = manejadorProyecto.actualizarProyecto(proyecto);
145           proyecto = manejadorProyecto.cargarUsuarios(proyecto);
            }else{
147           return LOGIN;
            }
149       }catch(Exception e){
            e.printStackTrace();
151         return ERROR;
          }
153       return SUCCESS;
      }
155
//    public String registrarUsuario(){
157 //      try{
//          /*
159 //          if( session instanceof SessionMap && ((SessionMap)session).get("
    idP") != null){
//            usuario = manejadorUsuario.add(usuario);
161 //
//            long idProyecto = (Long)((SessionMap)session).get("idP");
163 //            proyecto = manejadorProyecto.datosProyecto(idProyecto);
//            proyecto = manejadorProyecto.cargarUsuarios(proyecto);
165 //            proyecto.agregarPersonal(usuario);
//            proyecto = manejadorProyecto.actualizarProyecto(proyecto);
167 //          }else{
//            return LOGIN;
169 //          }*/
//      }catch(Exception e){
171 //          e.printStackTrace();
//          return ERROR;
173 //      }
//      return SUCCESS;
175 //    }

177
}
```

src/com/gestion/vista/personal/AgregarPersonalProyectoAction.java

```
package com.gestion.vista.personal;
2
import java.util.Iterator;
4 import java.util.List;
import java.util.Map;
6 import java.util.Set;
```

```
8 import org.apache.struts2.dispatcher.SessionMap;
  import org.apache.struts2.interceptor.SessionAware;
10
  import com.gestion.modelo.manejadorBd.proyecto.ManejadorProyecto;
12 import com.gestion.modelo.manejadorBd.rol.ManejadorRol;
  import com.gestion.modelo.manejadorBd.usuario.ManejadorUsuario;
14 import com.gestion.modelo.proyecto.Proyecto;
  import com.gestion.modelo.rol.Rol;
16 import com.gestion.modelo.usuario.Usuario;
  import com.opensymphony.xwork2.ActionSupport;
18
  public class NuevoPersonalAction extends ActionSupport implements
     SessionAware{
20   private Usuario usuario;
     private String password1;
22   private List<Rol> roles;
     private Set<String> rolesAsignados;
24
     private ManejadorUsuario manejadorUsuario;
26   private ManejadorRol manejadorRol;
     private ManejadorProyecto manejadorProyecto;
28   private Map session;

30   public NuevoPersonalAction(){
       manejadorUsuario = new ManejadorUsuario();
32     manejadorRol = new ManejadorRol();
       manejadorProyecto = new ManejadorProyecto();
34   }

36   public Usuario getUsuario() {
       return usuario;
38   }

40   public void setUsuario(Usuario usuario) {
       this.usuario = usuario;
42   }

44   public String getPassword1() {
       return password1;
46   }

48   public void setPassword1(String password1) {
       this.password1 = password1;
50   }
     public List<Rol> getRoles() {
52     return roles;
     }
54
     public void setRoles(List<Rol> roles) {
56     this.roles = roles;
     }
```

```
58
   public Set<String> getRolesAsignados() {
60   return rolesAsignados;
   }
62
   public void setRolesAsignados(Set<String> rolesAsignados) {
64   this.rolesAsignados = rolesAsignados;
   }
66
   @Override
68 public void setSession(Map<String, Object> arg0) {
     // TODO Auto-generated method stub
70   this.session = arg0;
   }
72
   @org.apache.struts2.interceptor.validation.SkipValidation
74 public String nuevoUsuario(){
     try{
76     roles = manejadorRol.lista();
       System.out.println("Roles: "+roles.size());
78   }catch(Exception e){
       e.printStackTrace();
80     return ERROR;
     }
82   return SUCCESS;
   }
84
   public String registrarUsuario(){
86   try{
       System.out.println(usuario.getIdUsuario());
88     System.out.println(usuario.getLogin());
       System.out.println(usuario.getNombreCompleto());
90
       System.out.println(usuario.getPass_word());
92     System.out.println(this.getPassword1());


94
       Iterator<String> it = rolesAsignados.iterator();
96


98     Rol a;
       while(it.hasNext()){
100      a = manejadorRol.find(it.next());
         if(a!=null){
102        usuario.addRol(a);
         }
104    }
       Proyecto proyecto;
106    if((session instanceof SessionMap) && (((SessionMap)session).get("
          idP")!=null)){
         long idProyecto = (Long)(((SessionMap)session).get("idP"));
108      proyecto = manejadorProyecto.datosProyecto(idProyecto);
```

```
        proyecto = manejadorProyecto.cargarUsuarios(proyecto);
        usuario = manejadorUsuario.add(usuario);
        proyecto.agregarPersonal(usuario);
        proyecto = manejadorProyecto.actualizarProyecto(proyecto);
      }else{
        return LOGIN;
      }

    }catch(Exception e){
      e.printStackTrace();
      return ERROR;
    }
    return SUCCESS;
  }

}
```

src/com/gestion/vista/personal/NuevoPersonalAction.java

```
package com.gestion.vista.personal;

import java.util.Map;

import org.apache.struts2.dispatcher.SessionMap;
import org.apache.struts2.interceptor.SessionAware;

import com.gestion.modelo.manejadorBd.proyecto.ManejadorProyecto;
import com.gestion.modelo.manejadorBd.usuario.ManejadorUsuario;
import com.gestion.modelo.proyecto.Proyecto;
import com.gestion.modelo.usuario.Usuario;
import com.opensymphony.xwork2.ActionSupport;

public class AdministrarPersonalAction extends ActionSupport implements
    SessionAware{
  private Proyecto proyecto;
  private Map session;
  private String idUsuario;
  private Usuario usuario;

  private ManejadorProyecto manejadorProyecto;
  private ManejadorUsuario manejadorUsuario;

  public AdministrarPersonalAction(){
    manejadorProyecto = new ManejadorProyecto();
    manejadorUsuario = new ManejadorUsuario();
  }

  public Proyecto getProyecto() {
    return proyecto;
  }

  public void setProyecto(Proyecto proyecto) {
    this.proyecto = proyecto;
```

```
34    }

36    public String getIdUsuario() {
        return idUsuario;
38    }

40    public void setIdUsuario(String idUsuario) {
        this.idUsuario = idUsuario;
42    }

44    public Usuario getUsuario() {
        return usuario;
46    }

48    public void setUsuario(Usuario usuario) {
        this.usuario = usuario;
50    }

52    public String personalProyecto(){
        try{
54        if((session instanceof SessionMap) && (((SessionMap)session).get("
              idP")!=null)){
            long idProyecto = (Long)(((SessionMap)session).get("idP"));
56          proyecto = manejadorProyecto.datosProyecto(idProyecto);
            proyecto = manejadorProyecto.cargarUsuarios(proyecto);
58        }else{
            System.out.println("Objecto session null");
60          return LOGIN;
          }
62      }catch(Exception e){
          e.printStackTrace();
64        this.addActionMessage("No se pudo obtener los usuarios del
              proyecto, asegurese de haber establecido un proyecto de
              trabajo.");
          return ERROR;
66      }
        return SUCCESS;
68    }
      /**
70     * Muestra los detalles del usuario cuyo identificador es idUsuario
       * @return
72     */
      public String detallesUsuario(){
74      try{
          System.out.println(this.getIdUsuario());
76        if(this.getIdUsuario() != null){
            usuario = manejadorUsuario.buscar(idUsuario);
78          usuario = manejadorUsuario.cargarRoles(usuario);
          }
80      }catch(Exception e){
          e.printStackTrace();
```

```
82        this.addActionMessage("No se pudieron obtener los datos del
              usuario");
          return ERROR;
84      }
        return SUCCESS;
86    }

88    @Override
      public void setSession(Map<String, Object> arg0) {
90      // TODO Auto-generated method stub
        this.session = arg0;
92    }

94  }
```

src/com/gestion/vista/personal/AdministrarPersonalAction.java

```
<!DOCTYPE validators PUBLIC
2       "-//OpenSymphony Group//XWork Validator 1.0.2//EN"
        "http://www.opensymphony.com/xwork/xwork-validator-1.0.2.dtd">
4   <validators>
        <field name="usuario.nombre">
6           <field-validator type="requiredstring">
            <param name="trim">true</param>
8               <message>Se necesita al menos el nombre</message>
            </field-validator>
10      </field>

12      <field name="usuario.correo">
            <field-validator type="requiredstring" short-circuit="true">
14            <param name="trim">true</param>
                <message>Se necesita un correo</message>
16          </field-validator>
            <field-validator type="email">
18              <message>Correo no valido</message>
            </field-validator>
20      </field>
    </validators>
```

src/com/gestion/vista/personal/ModificarEliminarUsuarioAction-validation.xml

```
1   package com.gestion.vista.tareasUsuario;

3   import java.util.ArrayList;
    import java.util.Iterator;
5   import java.util.List;
    import java.util.Map;
7
    import org.apache.struts2.dispatcher.SessionMap;
9   import org.apache.struts2.interceptor.SessionAware;

11  import com.gestion.modelo.manejadorBd.fase.ManejadorFase;
    import com.gestion.modelo.manejadorBd.fase.ManejadorTarea;
```

```
13 import com.gestion.modelo.manejadorBd.proyecto.ManejadorProyecto;
   import com.gestion.modelo.manejadorBd.usuario.ManejadorUsuario;
15 import com.gestion.modelo.proyecto.Fase;
   import com.gestion.modelo.proyecto.IdTarea;
17 import com.gestion.modelo.proyecto.Proyecto;
   import com.gestion.modelo.proyecto.Tarea;
19 import com.gestion.modelo.usuario.Usuario;
   import com.opensymphony.xwork2.ActionSupport;
21
   public class TareasUsuarioAction extends ActionSupport implements
       SessionAware{
23   private Map session;
     private Usuario usuario;
25   private List<Proyecto> proyectosUsuario;
     private List<Fase> fases;
27   private ManejadorUsuario manejadorUsuario;
     private ManejadorFase manejadorFase;
29   private ManejadorProyecto manejadorProyectos;

31   public TareasUsuarioAction(){
       manejadorUsuario = new ManejadorUsuario();
33     manejadorFase = new ManejadorFase();
       manejadorProyectos = new ManejadorProyecto();
35   }
     public Usuario getUsuario() {
37     return usuario;
     }
39   public void setUsuario(Usuario usuario) {
       this.usuario = usuario;
41   }
     public List<Proyecto> getProyectosUsuario() {
43     return proyectosUsuario;
     }
45   public void setProyectosUsuario(List<Proyecto> proyectosUsuario) {
       this.proyectosUsuario = proyectosUsuario;
47   }
     public List<Fase> getFases() {
49     return fases;
     }
51   public void setFases(List<Fase> fases) {
       this.fases = fases;
53   }

55   public String tareasUsuario(){
       try{
57       if((session instanceof SessionMap) && (((SessionMap)session).get("
             usuario")!=null)){
         usuario = (Usuario)((SessionMap)session).get("usuario");
59       usuario = manejadorUsuario.cargarProyectos(usuario);
         proyectosUsuario = manejadorProyectos.listaProyectos(usuario.
             getIdUsuario());
61       fases = manejadorFase.listaFases();
```

```
        usuario = manejadorUsuario.cargarTareasUsuario(usuario);
      }
    }catch(Exception e){
      e.printStackTrace();
      return ERROR;
    }
    return SUCCESS;
  }

  @Override
  public void setSession(Map<String, Object> arg0) {
    // TODO Auto-generated method stub
    this.session = arg0;
  }

}
```

src/com/gestion/vista/tareasUsuario/TareasUsuarioAction.java

```
package com.gestion.vista.tareasUsuario;

import org.apache.struts2.interceptor.SessionAware;

import com.gestion.modelo.manejadorBd.fase.ManejadorActividad;
import com.gestion.modelo.manejadorBd.fase.ManejadorTarea;
import com.gestion.modelo.manejadorBd.rol.ManejadorRol;
import com.gestion.modelo.proyecto.Actividad;
import com.gestion.modelo.proyecto.IdTarea;
import com.gestion.modelo.proyecto.Tarea;
import com.gestion.modelo.rol.Rol;
import com.opensymphony.xwork2.ActionSupport;

public class TareaUsuarioAction extends ActionSupport {
  private Tarea tarea;
  private long numProyecto;
  private long numIteracion;
  private long numTarea;
  private long idActividad;
  private Actividad actividad;
  private String nombreRol;
  private Rol rol;

  private ManejadorTarea manejadorTarea;
  private ManejadorActividad manejadorActividad;
  private ManejadorRol manejadorRol;

  public TareaUsuarioAction(){
    manejadorTarea = new ManejadorTarea();
    manejadorActividad = new ManejadorActividad();
    manejadorRol = new ManejadorRol();
  }

  public Tarea getTarea() {
```

```
35      return tarea;
    }
37
    public void setTarea(Tarea tarea) {
39      this.tarea = tarea;
    }
41
    public long getNumProyecto() {
43      return numProyecto;
    }
45
    public void setNumProyecto(long numProyecto) {
47      this.numProyecto = numProyecto;
    }
49
    public long getNumIteracion() {
51      return numIteracion;
    }
53
    public void setNumIteracion(long numIteracion) {
55      this.numIteracion = numIteracion;
    }
57
    public long getNumTarea() {
59      return numTarea;
    }
61
    public void setNumTarea(long numTarea) {
63      this.numTarea = numTarea;
    }
65
    public long getIdActividad() {
67      return idActividad;
    }
69
    public void setIdActividad(long idActividad) {
71      this.idActividad = idActividad;
    }
73
    public Actividad getActividad() {
75      return actividad;
    }
77
    public void setActividad(Actividad actividad) {
79      this.actividad = actividad;
    }
81
    public String getNombreRol() {
83      return nombreRol;
    }
85
    public void setNombreRol(String nombreRol) {
```

```
87      this.nombreRol = nombreRol;
    }
89
    public Rol getRol() {
91      return rol;
    }
93
    public void setRol(Rol rol) {
95      this.rol = rol;
    }
97
    public String detallesTareaUsuario(){
99      try{
          Actividad a;
101       tarea = manejadorTarea.datosTarea(new IdTarea(numProyecto,
              numIteracion, numTarea));
          a = tarea.getActividad();
103       a = manejadorActividad.cargarArtefactos(a);
        }catch(Exception e){
105       e.printStackTrace();
          return ERROR;
107     }
        return SUCCESS;
109   }

111   public String verDetallesActividad(){
        try{
113       actividad = manejadorActividad.buscarActividad(idActividad);
          actividad = manejadorActividad.cargarArtefactos(actividad);
115     }catch(Exception e){
          e.printStackTrace();
117       return ERROR;
        }
119     return SUCCESS;
    }
121
    public String verDetallesRol(){
123     try{
          rol = manejadorRol.find(nombreRol) ;
125     }catch(Exception e){
          e.printStackTrace();
127       return ERROR;
        }
129     return SUCCESS;
    }
131 }
```

src/com/gestion/vista/tareasUsuario/TareaUsuarioAction.java

```
1 package com.gestion.vista.tareasUsuario;

3 import java.io.File;
  import java.io.FileInputStream;
```

```
5  import java.io.IOException;
   import java.io.InputStream;
7  import java.sql.Blob;
   import java.sql.Date;
9  import java.util.Map;

11 import javax.servlet.ServletContext;
   import javax.servlet.http.HttpServletRequest;
13
   import org.apache.commons.io.FileUtils;
15 import org.apache.struts2.dispatcher.StreamResult;
   import org.apache.struts2.interceptor.ServletRequestAware;
17 import org.apache.struts2.interceptor.SessionAware;
   import org.apache.struts2.util.ServletContextAware;
19 import org.hibernate.Hibernate;

21 import com.gestion.modelo.ejemplarArtefacto.EjemplarArtefacto;
   import com.gestion.modelo.ejemplarArtefacto.IdEjemplarArtefacto;
23 import com.gestion.modelo.manejadorBd.ejemplarArtefacto.
       ManejadorEjemplarArtefacto;
   import com.gestion.modelo.manejadorBd.fase.ManejadorTarea;
25 import com.opensymphony.xwork2.ActionContext;
   import com.opensymphony.xwork2.ActionSupport;
27 import com.opensymphony.xwork2.Result;

29 public class SubirDescargarArtefactoAction extends ActionSupport
       implements ServletContextAware, ServletRequestAware{
     private Map session;
31   private long numProyecto;
     private long numIteracion;
33   private long numTarea;
     private File attachment;
35   private String attachmentFileName;
     private String nombreArtefacto;
37   private String nombreArchivo;
     private String attachmentContentType;
39   private String descripcionArchivo;
     private EjemplarArtefacto ejemplarArtefacto;
41   private ServletContext servletContext;
     private HttpServletRequest servletRequest;
43
     private ManejadorEjemplarArtefacto manejadorEjemplarArtefacto;
45   private ManejadorTarea manejadorTarea;

47   public SubirDescargarArtefactoAction(){
       manejadorEjemplarArtefacto = new ManejadorEjemplarArtefacto();
49     manejadorTarea = new ManejadorTarea();
     }
51
     public long getNumProyecto() {
53     return numProyecto;
     }
```

```
55
    public void setNumProyecto(long numProyecto) {
57    this.numProyecto = numProyecto;
    }
59
    public long getNumIteracion() {
61    return numIteracion;
    }
63
    public void setNumIteracion(long numIteracion) {
65    this.numIteracion = numIteracion;
    }
67
    public long getNumTarea() {
69    return numTarea;
    }
71
    public void setNumTarea(long numTarea) {
73    this.numTarea = numTarea;
    }
75
    public File getAttachment() {
77    return attachment;
    }
79
    public void setAttachment(File attachment) {
81    this.attachment = attachment;
    }
83
    public String getNombreArchivo() {
85    return nombreArchivo;
    }
87
    public void setNombreArchivo(String nombreArchivo) {
89    this.nombreArchivo = nombreArchivo;
    }
91
    public String getAttachmentContentType() {
93    return attachmentContentType;
    }
95
    public void setAttachmentContentType(String attachmentContentType) {
97    this.attachmentContentType = attachmentContentType;
    }
99
    public String getDescripcionArchivo() {
101   return descripcionArchivo;
    }
103
    public void setDescripcionArchivo(String descripcionArchivo) {
105   this.descripcionArchivo = descripcionArchivo;
    }
```

```
107
    public String getNombreArtefacto() {
109   return nombreArtefacto;
    }
111
    public void setNombreArtefacto(String nombreArtefacto) {
113   this.nombreArtefacto = nombreArtefacto;
    }
115
    public EjemplarArtefacto getEjemplarArtefacto() {
117   return ejemplarArtefacto;
    }
119
    public void setEjemplarArtefacto(EjemplarArtefacto ejemplarArtefacto)
       {
121   this.ejemplarArtefacto = ejemplarArtefacto;
    }
123
    public String getAttachmentFileName() {
125   return attachmentFileName;
    }
127
    public void setAttachmentFileName(String attachmentFileName) {
129   this.attachmentFileName = attachmentFileName;
    }
131
    public String subirEjemplar(){
133   try{
        System.out.println("Subiendo ejemplar ......");
135     EjemplarArtefacto ejemplar =  new EjemplarArtefacto();
        if( attachment != null){
137       IdEjemplarArtefacto id = new IdEjemplarArtefacto(numProyecto,
             nombreArtefacto);
          File copy = new File(attachmentFileName);
139

141          FileInputStream fio = new FileInputStream(attachment);
             Blob blob = Hibernate.createBlob(fio);
143          ejemplar.setContenidoArchivo(blob);
             ejemplar.setFechaCreacion(new Date(new java.util.Date().
                getTime()));
145          ejemplar.setNombreArchivo(attachmentFileName);
             ejemplar.setTipoContenido(attachmentContentType);
147          ejemplar.setIdEjemplar(id);
             ejemplar = manejadorEjemplarArtefacto.add(ejemplar);
149     }
      }catch(Exception e){
151     e.printStackTrace();
        return ERROR;
153   }
      return SUCCESS;
155 }
```

```
157   public String seleccionarArchivo(){
        try{
159
        }catch(Exception e){
161       e.printStackTrace();
          return ERROR;
163     }
        return SUCCESS;
165   }
      public InputStream getInputStream() throws Exception {
167     IdEjemplarArtefacto idEjemplar = new IdEjemplarArtefacto(numProyecto
            , nombreArtefacto);
        ejemplarArtefacto = manejadorEjemplarArtefacto.buscar(idEjemplar);
169     Result result = ActionContext.getContext().getActionInvocation().
            getResult();
        if( result != null && result instanceof StreamResult){
171       StreamResult streamResult = (StreamResult)result;
          streamResult.setContentDisposition(ejemplarArtefacto.
              getNombreArchivo());
173       streamResult.setContentType(ejemplarArtefacto.getTipoContenido());
          return ejemplarArtefacto.getContenidoArchivo().getBinaryStream();
175     }
        this.addActionMessage("No se encontro ninguno archivo");
177     return new InputStream() {

179       @Override
          public int read() throws IOException {
181         // TODO Auto-generated method stub
            return 0;
183       }
        };
185   }

187   @Override
      public void setServletContext(ServletContext arg0) {
189     // TODO Auto-generated method stub
        this.servletContext = arg0;
191   }

193   @Override
      public void setServletRequest(HttpServletRequest arg0) {
195     // TODO Auto-generated method stub
        this.servletRequest = arg0;
197   }

199 }
```

src/com/gestion/vista/tareasUsuario/SubirDescargarArtefactoAction.java

```
1 package com.gestion.vista.tareasUsuario;

3 import java.util.List;
```

```
import java.util.Set;

import com.gestion.modelo.manejadorBd.fase.ManejadorActividad;
import com.gestion.modelo.manejadorBd.fase.ManejadorTarea;
import com.gestion.modelo.proyecto.Actividad;
import com.gestion.modelo.proyecto.Artefacto;
import com.gestion.modelo.proyecto.IdTarea;
import com.gestion.modelo.proyecto.Tarea;
import com.opensymphony.xwork2.ActionSupport;

public class ArtefactosTareaAction extends ActionSupport{
  private long numProyecto;
  private long numIteracion;
  private long numTarea;
  private Set<Artefacto> artefactosRequeridos;
  private Set<Artefacto> artefactosProducidos;

  private ManejadorTarea manejadorTarea;
  private ManejadorActividad manejadorActividad;

  public ArtefactosTareaAction(){
    manejadorTarea = new ManejadorTarea();
    manejadorActividad = new ManejadorActividad();
  }
  public long getNumProyecto() {
    return numProyecto;
  }
  public void setNumProyecto(long numProyecto) {
    this.numProyecto = numProyecto;
  }
  public long getNumIteracion() {
    return numIteracion;
  }
  public void setNumIteracion(long numIteracion) {
    this.numIteracion = numIteracion;
  }
  public long getNumTarea() {
    return numTarea;
  }
  public void setNumTarea(long numTarea) {
    this.numTarea = numTarea;
  }
  public Set<Artefacto> getArtefactosRequeridos() {
    return artefactosRequeridos;
  }
  public void setArtefactosRequeridos(Set<Artefacto>
      artefactosRequeridos) {
    this.artefactosRequeridos = artefactosRequeridos;
  }
  public Set<Artefacto> getArtefactosProducidos() {
    return artefactosProducidos;
  }
```

```
55    public void setArtefactosProducidos(Set<Artefacto>
          artefactosProducidos) {
        this.artefactosProducidos = artefactosProducidos;
57    }
      public String artefactosTarea(){
59      try{
          IdTarea idTarea = new IdTarea(numProyecto, numIteracion, numTarea)
              ;
61        Tarea a = manejadorTarea.datosTarea(idTarea);
          Actividad actividad = a.getActividad();
63        actividad = manejadorActividad.cargarArtefactos(actividad);
          artefactosRequeridos = actividad.getRequeridos();
65        artefactosProducidos = actividad.getProducidos();
        }catch(Exception e){
67        e.printStackTrace();
          return ERROR;
69      }
        return SUCCESS;
71    }
  }
```

src/com/gestion/vista/tareasUsuario/ArtefactosTareaAction.java

```
  package com.gestion.vista.planificacion;
2
  import java.util.ArrayList;
4 import java.util.HashSet;
  import java.util.Iterator;
6 import java.util.List;
  import java.util.Map;
8 import java.util.Set;

10 import org.apache.struts2.dispatcher.SessionMap;
   import org.apache.struts2.interceptor.SessionAware;
12
   import com.gestion.modelo.manejadorBd.fase.ManejadorActividad;
14 import com.gestion.modelo.manejadorBd.fase.ManejadorDisciplina;
   import com.gestion.modelo.manejadorBd.fase.ManejadorIteracion;
16 import com.gestion.modelo.manejadorBd.fase.ManejadorTarea;
   import com.gestion.modelo.manejadorBd.usuario.ManejadorUsuario;
18 import com.gestion.modelo.proyecto.Actividad;
   import com.gestion.modelo.proyecto.Disciplina;
20 import com.gestion.modelo.proyecto.IdIteracion;
   import com.gestion.modelo.proyecto.IdTarea;
22 import com.gestion.modelo.proyecto.Iteracion;
   import com.gestion.modelo.proyecto.Tarea;
24 import com.gestion.modelo.usuario.Usuario;
   import com.gestion.vista.json.ActividadJson;
26 import com.gestion.vista.json.TareaJson;
   import com.gestion.vista.json.UsuarioJson;
28 import com.opensymphony.xwork2.ActionSupport;
```

```
30 public class NuevaTareaAction extends ActionSupport implements
     SessionAware{
   private long numIteracion;
32 private long disciplinaElegida;
   private long actividadElegida;
34 private long padreElegido;
   private Tarea tarea;
36 private boolean pOpen;
   private boolean hito;
38 private boolean pGroup;

40 private List<ActividadJson> actividadesJson;
   private List<Actividad> actividades;
42 private List<UsuarioJson> usuariosJson;
   private List<TareaJson> candidatasPadre;
44 private List<Disciplina> disciplinas;
   private List<String> responsables;
46 private List<String> tareasPredecesoras;
   private List<TareaJson> predecesoresCandidatosJson;
48
   private ManejadorActividad manejadorActividad;
50 private ManejadorTarea manejadorTarea;
   private ManejadorIteracion manejadorIteracion;
52 private ManejadorDisciplina manejadorDisciplina;
   private ManejadorUsuario manejadorUsuario;
54
   private Map session;
56
   public NuevaTareaAction(){
58   manejadorActividad = new ManejadorActividad();
     manejadorTarea = new ManejadorTarea();
60   manejadorIteracion = new ManejadorIteracion();
     manejadorDisciplina = new ManejadorDisciplina();
62   manejadorUsuario = new ManejadorUsuario();
   }
64 public long getDisciplinaElegida() {
     return disciplinaElegida;
66 }
   public void setDisciplinaElegida(long disciplinaElegida) {
68   this.disciplinaElegida = disciplinaElegida;
   }
70 public long getActividadElegida() {
     return actividadElegida;
72 }
   public void setActividadElegida(long actividadElegida) {
74   this.actividadElegida = actividadElegida;
   }
76 public List<Actividad> getActividades() {
     return actividades;
78 }
   public void setActividades(List<Actividad> actividades) {
80   this.actividades = actividades;
```

```
  }
  public List<ActividadJson> getActividadesJson() {
    return actividadesJson;
  }
  public void setActividadesJson(List<ActividadJson> actividadesJson) {
    this.actividadesJson = actividadesJson;
  }
  public List<UsuarioJson> getUsuariosJson() {
    return usuariosJson;
  }
  public void setUsuariosJson(List<UsuarioJson> usuariosJson) {
    this.usuariosJson = usuariosJson;
  }

  public long getNumIteracion() {
    return numIteracion;
  }
  public void setNumIteracion(long numIteracion) {
    this.numIteracion = numIteracion;
  }
  public List<TareaJson> getCandidatasPadre() {
    return candidatasPadre;
  }
  public void setCandidatasPadre(List<TareaJson> candidatasPadre) {
    this.candidatasPadre = candidatasPadre;
  }
  public Tarea getTarea() {
    return tarea;
  }
  public void setTarea(Tarea tarea) {
    this.tarea = tarea;
  }
  public long getPadreElegido() {
    return padreElegido;
  }
  public void setPadreElegido(long padreElegido) {
    this.padreElegido = padreElegido;
  }

  public List<Disciplina> getDisciplinas() {
    return disciplinas;
  }
  public void setDisciplinas(List<Disciplina> disciplinas) {
    this.disciplinas = disciplinas;
  }
  public List<String> getResponsables() {
    return responsables;
  }
  public void setResponsables(List<String> responsables) {
    this.responsables = responsables;
  }
  public List<String> getTareasPredecesoras() {
```

```
    return tareasPredecesoras;
  }
  public void setTareasPredecesoras(List<String> tareasPredecesoras) {
    this.tareasPredecesoras = tareasPredecesoras;
  }
  public boolean ispOpen() {
    return pOpen;
  }
  public void setpOpen(boolean pOpen) {
    this.pOpen = pOpen;
  }
  public boolean isHito() {
    return hito;
  }
  public void setHito(boolean hito) {
    this.hito = hito;
  }
  public boolean ispGroup() {
    return pGroup;
  }
  public void setpGroup(boolean pGroup) {
    this.pGroup = pGroup;
  }

  public List<TareaJson> getPredecesoresCandidatosJson() {
    return predecesoresCandidatosJson;
  }
  public void setPredecesoresCandidatosJson(
      List<TareaJson> predecesoresCandidatosJson) {
    this.predecesoresCandidatosJson = predecesoresCandidatosJson;
  }
  /**
   * se llama con el parametro numIteracion
   * @return
   */
  public String nuevaTarea(){
    try{
      /**
       * Obtener el id del proyecto de session
       */
      disciplinas = manejadorDisciplina.lista();

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

    return "success";
  }


  public String getJSONActividadesDisciplina(){
    actividades = manejadorActividad.lista(disciplinaElegida);
```

```
    Iterator<Actividad> it = actividades.iterator();
186 actividadesJson = new ArrayList<ActividadJson>();
    while(it.hasNext()){
188   actividadesJson.add(new ActividadJson(it.next()));
    }
190 /**
     * numero de proyecto obtenido de sesion
192  */
    System.out.println("Num iteracion: "+numIteracion);
194 Long idp = (Long)((SessionMap)session).get("idP");
    if(idp==null){
196   idp=new Long(-1);
    }
198
    IdIteracion id = new IdIteracion(idp, numIteracion);
200 List<Tarea> tareas = manejadorIteracion.tareasIteracion(id);
    Iterator<Tarea> it2 = tareas.iterator();
202 Tarea aux;
    candidatasPadre = new ArrayList<TareaJson>();
204 while(it2.hasNext()){
      aux = it2.next();
206   aux = manejadorTarea.cargarUsuariosTareasPrecedentes(aux);

208   if(aux.getActividad().getDisciplina().getIdDisciplina()==
        disciplinaElegida)
      candidatasPadre.add(new TareaJson(aux));
210 }

212 actividades = null;
    return "success";
214 }
    /**
216  * Crea una lista de tareas del tipo TareaJson
     * que son subtareas de la tarea identificada por
218  * 'padreElegido' y la iteracion 'numIteracion'
     * el valor de idProyecto se obtendra de la sesion.
220  * @return
     */
222 public String getJSONsubtareas(){
      IdTarea id = new IdTarea(1, numIteracion, padreElegido);
224   List<Tarea> subtareas = manejadorTarea.subtareasDeTarea(id);
      predecesoresCandidatosJson = new ArrayList<TareaJson>();
226   Iterator<Tarea> it = subtareas.iterator();

228   Tarea a;
      while(it.hasNext()){
230     a = it.next();
        a = manejadorTarea.cargarUsuariosTareasPrecedentes(a);
232     predecesoresCandidatosJson.add(new TareaJson(a));
      }
234
      return "success";
```

```
236   }

238   public String guardarTarea(){
        try{
240       long idProyecto = this.idProyectoActual();

242       if( idProyecto < 1) return ERROR;

244       IdIteracion idIter = new IdIteracion(idProyecto, numIteracion);
          Iteracion iteracion = manejadorIteracion.buscarIteracion(idIter);
246       iteracion = manejadorIteracion.cargarTareas(iteracion);
          /**
248        * Creamos el nuevo IdTarea y asignamos
           */
250       IdTarea idNueva = new IdTarea(idProyecto, numIteracion,
             manejadorTarea.numTareaMax(idProyecto, numIteracion)+1);
          tarea.setIdTarea(idNueva);
252
          /**
254        * Recuperamos actividad y asignamos
           */
256       Actividad act = manejadorActividad.buscarActividad(this.
             getActividadElegida());
          tarea.setActividad(act);
258
          /**
260        * Asignamos la iteracion a la que pertenece, lo cual no se si es
              necesario
           */
262       tarea.setIteracion(iteracion);

264       /**
           * Creamos un instancia de tarea
266        */
          Tarea nueva = manejadorTarea.agregarTarea(tarea);
268
          /**
270        * Recuperamos tarea padre y asignamos
           *
272        */
          Tarea padre =null;
274       if( this.getPadreElegido() > 0 ){
            IdTarea idTarea = new IdTarea(idProyecto, numIteracion, this.
               getPadreElegido());
276         padre = manejadorTarea.datosTarea(idTarea);

278         if( padre !=null){
              padre.setpGroup(1);
280           padre.setpOpen(1);

282
              List<Tarea> padres = new ArrayList<Tarea>();
```

```
284
            padres.add(padre);
286         nueva.setTareaPadre(padres);
          }else{
288         System.out.println("Padre es null");
          }
290     }else{
          System.out.println("No entro al bucle "+this.getPadreElegido());
292     }


294
      //nueva = manejadorTarea.datosTarea(idNueva);
296
      /**
298    * Creamos las lista de predecesores y asignamos
       */
300   List<Tarea> predecesores = new ArrayList<Tarea>();
      Iterator<String> it = this.getTareasPredecesoras().iterator();
302
      Tarea p;
304
      while(it.hasNext()){
306     IdTarea idp = new IdTarea(idProyecto, numIteracion, Long.
            parseLong(it.next()));
        p = manejadorTarea.datosTarea(idp);
308
        if(p!=null){
310       if(padre != null)
            if(p.getIdTarea().getNumTarea() == padre.getIdTarea().
                getNumTarea()) continue;
312       if( p.getFechaFin().after(nueva.getFechaInicio())){
            nueva.setFechaInicio(p.getFechaFin());
314       }
          predecesores.add(p);
316     }
      }
318
      nueva.setPredecesores(predecesores);
320   //nueva = manejadorTarea.actualizarTarea(nueva);


322
      /**
324    * Recuperamos recursos y asignamos
       */
326
      Set<Usuario> usuariosAsignados = new HashSet<Usuario>();
328   it = this.getResponsables().iterator();
      Usuario u;
330
      while(it.hasNext()){
332     u = manejadorUsuario.buscar(it.next());
        if(u != null){
```

```
334             usuariosAsignados.add(u);
                }
336         }

338         if( nueva.getFechaInicio().after(nueva.getFechaFin())){
              nueva.setFechaFin(nueva.getFechaInicio());
340         }
            nueva.setRecursos(usuariosAsignados);
342         tarea = manejadorTarea.actualizarTarea(nueva);


344
            System.out.println("IdTArea: "+tarea.getIdTarea().toString());
346         System.out.println("Tarea: "+tarea.getNombreTarea());
            System.out.println("Fecha de inicio: "+tarea.getFechaInicio());
348         System.out.println("Fecha de termino: "+tarea.getFechaFin());
            System.out.println("Porcentaje: "+tarea.getPorcentaje());
350
            System.out.println("Actividad asociada: "+tarea.getActividad().
                getNombre());
352         if( tarea.getTareaPadre().size()>0)
              System.out.println("Tarea padre: "+tarea.getTareaPadre().get(0).
                  getIdTarea().getIdCadena());
354         System.out.println("Predecesores: "+ tarea.getPredecesores().
                toString());
            System.out.println("Usuarios asignados: "+tarea.getRecursos().
                toString());
356
            System.out.println("Actualizando tarea....");
358
            System.out.println("Finalizado");
360         if(padre != null){
              padre = manejadorTarea.actualizarTarea(padre);
362         }
        }catch(Exception e){
364       e.printStackTrace();
          return ERROR;
366     }
        return "success";
368   }
    public long idProyectoActual(){
370     long idProyecto;
        if((session instanceof SessionMap) && (((SessionMap)session).get("
            idP")!=null)){
372         idProyecto = (Long)(((SessionMap)session).get("idP"));
            return idProyecto;
374     }else{
          return 0;
376     }
    }
378 @Override
    public void setSession(Map<String, Object> arg0) {
380   // TODO Auto-generated method stub
```

```
    this.session = arg0;
  }
}
```

src/com/gestion/vista/planificacion/NuevaTareaAction.java

```
package com.gestion.vista.planificacion;

import java.util.List;
import java.util.Map;

import org.apache.struts2.dispatcher.SessionMap;
import org.apache.struts2.interceptor.SessionAware;

import com.gestion.modelo.manejadorBd.fase.ManejadorFase;
import com.gestion.modelo.manejadorBd.fase.ManejadorIteracion;
import com.gestion.modelo.proyecto.Fase;
import com.gestion.modelo.proyecto.IdIteracion;
import com.gestion.modelo.proyecto.Iteracion;
import com.gestion.modelo.proyecto.Proyecto;
import com.gestion.modelo.proyecto.Tarea;
import com.opensymphony.xwork2.ActionSupport;

public class IteracionAction extends ActionSupport implements
    SessionAware{
  private ManejadorIteracion manejadorIteracion;
  private ManejadorFase manejadorFase;
  private Iteracion iteracion;
  private List<Iteracion> iteraciones;
  private List<Tarea> tareas;
  private int numIteracion;
  private int tareaElegida;
  private List<Fase> fases;
  private int faseElegida;

  Map session;

  public IteracionAction(){
    manejadorIteracion = new ManejadorIteracion();
    manejadorFase = new ManejadorFase();
  }

  public Iteracion getIteracion() {
    return iteracion;
  }
  public void setIteracion(Iteracion iteracion) {
    this.iteracion = iteracion;
  }

  public List<Iteracion> getIteraciones() {
    return iteraciones;
  }
```

```
47    public void setIteraciones(List<Iteracion> iteraciones) {
        this.iteraciones = iteraciones;
49    }

51    public List<Tarea> getTareas() {
        return tareas;
53    }

55    public void setTareas(List<Tarea> tareas) {
        this.tareas = tareas;
57    }

59    public int getNumIteracion() {
        return numIteracion;
61    }

63    public void setNumIteracion(int numIteracion) {
        this.numIteracion = numIteracion;
65    }

67    public int getTareaElegida() {
        return tareaElegida;
69    }

71    public void setTareaElegida(int tareaElegida) {
        this.tareaElegida = tareaElegida;
73    }

75    public List<Fase> getFases() {
        return fases;
77    }

79    public void setFases(List<Fase> fases) {
        this.fases = fases;
81    }

83    public int getFaseElegida() {
        return faseElegida;
85    }

87    public void setFaseElegida(int faseElegida) {
        this.faseElegida = faseElegida;
89    }

91    /*
      public String agregarIteracion(){
93      try{
          System.out.println("Agregando Iteracion");
95        IdIteracion id = new IdIteracion();
          id.setIdProyecto(4);
97        id.setFase("Inicio");
          id.setNumIteracion(3);
```

```
99          this.getIteracion().setId(id);
            iteracion = manejadorIteracion.agregarIteracion(this.getIteracion
                ());
101         iteraciones = manejadorIteracion.iteracionesProyecto(4, id.getFase
                ());
        }catch(Exception e){
103       e.printStackTrace();
        }
105     return "success";
    }
107 */
    /**
109  * Obtiene las iteraciones del proyecto actual.
     */
111 public String iteraciones(){
      try{
113     long idProyecto;
        if((session instanceof SessionMap) && (((SessionMap)session).get("
            idP")!=null)){
115       idProyecto = (Long)(((SessionMap)session).get("idP"));
          iteraciones = manejadorIteracion.iteracionesProyecto(idProyecto
              );
117       System.out.println("Iteraciones del proyecto: "+idProyecto);
        }else{
119       return ERROR;
        }
121   }catch(Exception e){
        e.printStackTrace();
123     return ERROR;
      }
125   return SUCCESS;
    }
127 /**
     * Obtiene las Tareas de la iteracion y proyecto actual.
129  * @return
     */
131 public String tareas(){
      System.out.println("Obteniendo tareas..... clase
          IteracionAction");
133   try{
        long idProyecto = this.idProyectoActual();
135     if( idProyecto >0 ){
          IdIteracion id = new IdIteracion();
137       id.setIdProyecto(idProyecto);
          id.setNumIteracion(this.getNumIteracion());
139     }else{
          return ERROR;
141     }
      }catch(Exception e){
143     e.printStackTrace();
        return ERROR;
145   }
```

```
    return "success";
147 }
    /**
149  * Prepara la lista de fases que se necesita al momento de agregar
     * una nueva iteracion.
151  * @return
     */
153 public String nuevaIteracion(){
      try{
155     fases = manejadorFase.listaFases();
      }catch(Exception e){
157     e.printStackTrace();
        return ERROR;
159   }
      return "success";
161 }
    /**
163  * Agrega una iteracion
     * @return
165  */
    public String agregarIteracion(){
167   try{
        /**
169      * Solicitamos el numIteracion mas grande
         * que existe en la base de datos para el proyecto 7
171      * y fase 1
         */
173     long idProyecto;
        if((session instanceof SessionMap) && (((SessionMap)session).get("
           idP")!=null)){
175       idProyecto = (Long)(((SessionMap)session).get("idP"));
        }else{
177       return ERROR;
        }
179

181     Integer numIter = manejadorIteracion.getMaxNumIteracion(idProyecto
           );
        if( numIter == null) numIter = 0;
183     /**
         * creamos y asignamos el identificador
185      */
        System.out.println("Id proyecto: "+idProyecto);
187     IdIteracion id = new IdIteracion();
        id.setIdProyecto(idProyecto);
189     id.setNumIteracion((long)(numIter+1));
        this.getIteracion().setId(id);
191

        /**
193      * Asociamos con la fase elegida
         */
195     Fase fase = manejadorFase.fase(this.faseElegida);
```

```
        this.getIteracion().setFase(fase);
197
        /**
199      * agregamos a la base de datos
         */
201     iteracion = manejadorIteracion.agregarIteracion(this.getIteracion
            ());

203     /**
         * cargamos las iteraciones para la vista iteraciones
205      */
        iteraciones = manejadorIteracion.iteracionesProyecto(idProyecto);
207   }catch(Exception e){
        System.out.println("Error en agregarIteracion");
209     e.printStackTrace();
        return ERROR;
211   }
      return "success";
213 }
    /**
215  * Prepara los datos para modificar o eliminar una iteracion
     * @return
217  */
    public String modificarEliminarIteracion(){
219   try{
        long idP = this.idProyectoActual();
221     if( idP>0){
          IdIteracion id = new IdIteracion(idP, this.getNumIteracion());
223       iteracion = manejadorIteracion.buscarIteracion(id);
          fases = manejadorFase.listaFases();
225     }else{
          return ERROR;
227     }
      }catch(Exception e){
229     e.printStackTrace();
        return ERROR;
231   }
      return SUCCESS;
233 }
    /**
235  * Elimina la iteracion elegida
     * @return
237  */
    public String eliminarIteracion(){
239   try{
        long idProyecto = this.idProyectoActual();
241     if( idProyecto > 0){
          IdIteracion id = new IdIteracion();
243       id.setIdProyecto(idProyecto);
          id.setNumIteracion(this.getNumIteracion());
245       iteracion = manejadorIteracion.eliminarIteracion(id);
```

```
        iteraciones = manejadorIteracion.iteracionesProyecto(idProyecto)
            ;
      }else{
        return ERROR;
      }
    }catch(Exception e){
      e.printStackTrace();
      return ERROR;
    }
    return "success";
  }

  public String guardarCambiosIteracion(){
    try{
      Fase a = manejadorFase.fase(faseElegida);
      iteracion.setFase(a);
      iteracion = manejadorIteracion.actualizarIteracion(iteracion);
    }catch(Exception e){
      e.printStackTrace();
      return ERROR;
    }
    return SUCCESS;
  }

  @Override
  public void setSession(Map<String, Object> arg0) {
    // TODO Auto-generated method stub
    this.session = arg0;
  }

  public long idProyectoActual(){
    long idProyecto;
    if((session instanceof SessionMap) && (((SessionMap)session).get("
        idP")!=null)){
      idProyecto = (Long)(((SessionMap)session).get("idP"));
      return idProyecto;
    }else{
      return 0;
    }
  }
}
```

src/com/gestion/vista/planificacion/IteracionAction.java

```
package com.gestion.vista.planificacion;

import java.util.List;
import java.util.Map;

import org.apache.struts2.dispatcher.SessionMap;
import org.apache.struts2.interceptor.SessionAware;

import com.gestion.modelo.manejadorBd.fase.ManejadorTarea;
```

```
10 import com.gestion.modelo.proyecto.IdTarea;
   import com.gestion.modelo.proyecto.Tarea;
12 import com.opensymphony.xwork2.ActionSupport;

14 public class EliminarTareaAction extends ActionSupport{
     private long numIteracion;
16   private long numTarea;
     private long idProyecto;
18   private Tarea tarea;

20   private ManejadorTarea manejadorTarea;


22
     public EliminarTareaAction(){
24     manejadorTarea = new ManejadorTarea();
     }
26
     public long getNumIteracion() {
28     return numIteracion;
     }
30
     public void setNumIteracion(long numIteracion) {
32     this.numIteracion = numIteracion;
     }
34
     public long getNumTarea() {
36     return numTarea;
     }
38
     public void setNumTarea(long numTarea) {
40     this.numTarea = numTarea;
     }
42
     public long getIdProyecto() {
44     return idProyecto;
     }
46
     public void setIdProyecto(long idProyecto) {
48     this.idProyecto = idProyecto;
     }
50


52   public Tarea getTarea() {
       return tarea;
54   }

56   public void setTarea(Tarea tarea) {
       this.tarea = tarea;
58   }

60   public String eliminarTarea(){
       try{
```

```
62          IdTarea idTarea = new IdTarea(this.getIdProyecto(), numIteracion,
                numTarea);
            /**
64           * Verificar que no sea la unica subtarea
             */
66
            tarea = manejadorTarea.datosTarea(idTarea);
68          if( tarea.getTareaPadre().size()>0){
              Tarea padre = tarea.getTareaPadre().get(0);
70            padre = manejadorTarea.cargarSubtareas(padre);
              List<Tarea> subtareas = padre.getSubtareas();
72            if( subtareas.size() > 1){
                tarea = manejadorTarea.eliminarTarea(tarea.getIdTarea());
74            }else{
                padre.setpGroup(0);
76              padre.setpOpen(0);
                padre = manejadorTarea.actualizarTarea(padre);
78            }
            }
80          tarea = manejadorTarea.eliminarTarea(idTarea);
        }catch(Exception e){
82        e.printStackTrace();
        }
84      return "success";
    }
86  }
```

src/com/gestion/vista/planificacion/EliminarTareaAction.java

```
    package com.gestion.vista.planificacion;
2
    import java.util.ArrayList;
4   import java.util.HashSet;
    import java.util.Iterator;
6   import java.util.List;
    import java.util.Map;
8   import java.util.Set;

10  import org.apache.struts2.dispatcher.SessionMap;
    import org.apache.struts2.interceptor.SessionAware;
12
    import com.gestion.modelo.manejadorBd.fase.ManejadorActividad;
14  import com.gestion.modelo.manejadorBd.fase.ManejadorDisciplina;
    import com.gestion.modelo.manejadorBd.fase.ManejadorIteracion;
16  import com.gestion.modelo.manejadorBd.fase.ManejadorTarea;
    import com.gestion.modelo.manejadorBd.proyecto.ManejadorProyecto;
18  import com.gestion.modelo.proyecto.Actividad;
    import com.gestion.modelo.proyecto.Disciplina;
20  import com.gestion.modelo.proyecto.IdIteracion;
    import com.gestion.modelo.proyecto.Iteracion;
22  import com.gestion.modelo.proyecto.Proyecto;
    import com.gestion.modelo.proyecto.Tarea;
24  import com.gestion.modelo.usuario.Usuario;
```

```
import com.gestion.vista.json.TareaJson;
import com.opensymphony.xwork2.ActionSupport;

public class TareaAction extends ActionSupport implements SessionAware{
  private Proyecto proyecto;
  private Set<Usuario> usuariosProyecto = new HashSet<Usuario>();
  private Set<Usuario> usuariosAsignados = new HashSet<Usuario>();
  private Tarea tarea;
  private List<Disciplina> disciplinas = new ArrayList<Disciplina>();
  private List<Tarea> tareas = new ArrayList<Tarea>();
  private List<Actividad> actividades = new ArrayList<Actividad>();
  List<TareaJson> tareasJson;

  private long idTareaElegida;
  private long idProyecto;
  private long numIteracion;
  /*
   * Contendra la lista de CURP, de los usuarios
   * que se elijan como responsables de la nueva tarea.
   */
  private List<String> noresponsables;
  private List<String> responsables;


  private Map session;

  private ManejadorProyecto manejadorProyecto;
  private ManejadorIteracion manejadorIteracion;
  private ManejadorDisciplina manejadorDisciplina;
  private ManejadorActividad manejadorActividad;
  private ManejadorTarea manejadorTarea;

  public TareaAction(){
    manejadorTarea = new ManejadorTarea();
    manejadorIteracion = new ManejadorIteracion();
    manejadorProyecto =  new ManejadorProyecto();
    manejadorDisciplina = new ManejadorDisciplina();
    manejadorActividad = new ManejadorActividad();
  }
  public Proyecto getProyecto() {
    return proyecto;
  }
  public void setProyecto(Proyecto proyecto) {
    this.proyecto = proyecto;
  }
  public Set<Usuario> getUsuariosProyecto() {
    return usuariosProyecto;
  }
  public void setUsuariosProyecto(Set<Usuario> usuariosProyecto) {
    this.usuariosProyecto = usuariosProyecto;
  }
  public Tarea getTarea() {
```

```
    return tarea;
  }
  public void setTarea(Tarea tarea) {
    this.tarea = tarea;
  }
  public List<Tarea> getTareas() {
    return tareas;
  }
  public void setTareas(List<Tarea> tareas) {
    this.tareas = tareas;
  }
  public List<String> getResponsables() {
    return responsables;
  }
  public void setResponsables(List<String> responsables) {
    this.responsables = responsables;
  }
  public Set<Usuario> getUsuariosAsignados() {
    return usuariosAsignados;
  }
  public void setUsuariosAsignados(Set<Usuario> usuariosAsignados) {
    this.usuariosAsignados = usuariosAsignados;
  }
  public List<String> getNoresponsables() {
    return noresponsables;
  }
  public void setNoresponsables(List<String> noresponsables) {
    this.noresponsables = noresponsables;
  }

  public List<Disciplina> getDisciplinas() {
    return disciplinas;
  }
  public void setDisciplinas(List<Disciplina> disciplinas) {
    this.disciplinas = disciplinas;
  }
  public List<Actividad> getActividades() {
    return actividades;
  }
  public void setActividades(List<Actividad> actividades) {
    this.actividades = actividades;
  }

  public List<TareaJson> getTareasJson() {
    return tareasJson;
  }
  public void setTareasJson(List<TareaJson> tareasJson) {
    this.tareasJson = tareasJson;
  }
  public long getIdTareaElegida() {
    return idTareaElegida;
  }
```

```
  public void setIdTareaElegida(long idTareaElegida) {
130   this.idTareaElegida = idTareaElegida;
  }
132
  public long getIdProyecto() {
134   return idProyecto;
  }
136 public void setIdProyecto(long idProyecto) {
    this.idProyecto = idProyecto;
138 }
  public long getNumIteracion() {
140   return numIteracion;
  }
142 public void setNumIteracion(long numIteracion) {
    this.numIteracion = numIteracion;
144 }
  public String nuevaTarea(){
146   try{
      long idPtActual = this.idProyectoActual();
148     if( idPtActual > 0){
        disciplinas = manejadorDisciplina.lista();
150     actividades = manejadorActividad.lista(idPtActual);
        proyecto = manejadorProyecto.datosProyecto(idPtActual);
152     proyecto = manejadorProyecto.cargarUsuarios(proyecto);
        usuariosProyecto = proyecto.getListaUsuarios();
154
        IdIteracion id = new IdIteracion();
156     id.setIdProyecto(idPtActual);
        id.setNumIteracion(this.numIteracion);
158     Iteracion iteracion = manejadorIteracion.buscarIteracion(id);
        iteracion = manejadorIteracion.cargarTareas(iteracion);
160     tareas = iteracion.getTareas();

162     }else{
        return ERROR;
164     }
    }catch(Exception e ){
166   e.printStackTrace();
      //return "error";
168 }

170   return "success";
  }
172
  public String probarActividad(){
174   try{

176   }catch(Exception e){
      e.printStackTrace();
178 }
    return "success";
180 }
```

```
  /**
182  * Prepara la lista de Tareas a mostrar
   * @return
184  */
  public String tareas(){
186   try{
        long idPtActual = this.idProyectoActual();
188     if( idPtActual > 0){
          IdIteracion id = new IdIteracion(idPtActual, this.
              getNumIteracion());
190       Iteracion iter = new Iteracion();
          iter.setId(id);
192       this.obtenerListaMostrar(iter);
        }
194     else{
          return ERROR;
196     }
      }catch(Exception e){
198     e.printStackTrace();
        return ERROR;
200   }
      return SUCCESS;
202 }

204   public String getJSON(){
        return tareas();
206   }

208   public String detallesTarea(){
        nuevaTarea();
210     return "success";
      }
212
  public List<TareaJson> obtenerListaMostrar(Iteracion iteracion){
214   tareasJson = new ArrayList<TareaJson>();
      List<Disciplina> disciplinas = new ArrayList<Disciplina>();
216   List<Tarea> tareasSinOrden = new ArrayList<Tarea>();
      List<Tarea> tareas = new ArrayList<Tarea>();
218   ManejadorDisciplina md= new ManejadorDisciplina();

220   /**
       * Llenamos la lista de tareas de la iteracion
222    */
      tareasSinOrden = manejadorIteracion.tareasIteracion(iteracion.getId
          ());
224   System.out.println("Se han obtenido: "+tareasSinOrden.size() + "
          tareas");

226   Iterator<Tarea> it1 = tareasSinOrden.iterator();
      while(it1.hasNext()){
228     System.out.println("Tarea>>>> : "+it1.next().getNombreTarea());
      }
```

```
230
232     this.crearListas(tareasSinOrden, tareas);
        System.out.println("La lista ordenada se ha creado");
234     it1 = tareas.iterator();
        while(it1.hasNext()){
236       Tarea t = it1.next();
          System.out.println("Tarea>>>> : "+t.getNombreTarea()+"  "+t.
              getIdTarea().getIdProyecto());
238     }

240

        /**
242      * Llenamos la lista de disciplinas
         */
244     disciplinas = md.lista();

246     /**
         * Recorremos cada disciplina
248      */
        Iterator<Disciplina> it = disciplinas.iterator();
250     Iterator<Tarea> itt = null;
        Disciplina disciplina;
252     Tarea a;
        while(it.hasNext()){
254       disciplina = it.next();
          tareasJson.add(new TareaJson(disciplina));
256       //System.out.println("Disciplina: "+disciplina.getNombre());
          /**
258        * Iteramos sobre tareas
           */
260       itt = tareas.iterator();
          System.out.println("Disciplina actual: "+disciplina.
              getIdDisciplina());
262       while(itt.hasNext()){
            a = itt.next();
264         a = manejadorTarea.cargarUsuariosTareasPrecedentes(a);
            System.out.println("Disciplina de tarea: "+a.getActividad().
                getDisciplina().getIdDisciplina());
266

            if( a.getActividad().getDisciplina().getIdDisciplina().compareTo
                (disciplina.getIdDisciplina())==0){
268           System.out.println("Creando JSON para : "+a.getNombreTarea());
              tareasJson.add(new TareaJson(a));
270         }
          }
272

        }
274     return tareasJson;
    }
276

    /**
```

```
278     * Metodo recursivo para crear la lista de tareas
        * de forma ordenada.
280     * @param tareas
        * @param dibujar
282     */
    public void crearListas(List<Tarea> tareas, List<Tarea> dibujar){
284   if( tareas.size()>0){
        Tarea a = tareas.remove(0);
286     Tarea p = null;
        boolean band = false;
288     if(a.getTareaPadre().size()>0){
          p = a.getTareaPadre().get(0);
290       if(!dibujar.contains(p)) band = true;
        }
292
          if(dibujar.contains(a)){
294         System.out.println("La tarea: "+a.getNombreTarea()+" ya esta")
                ;
            crearListas(tareas,dibujar);
296       }else if(band){
            tareas.set(0, p);
298         crearListas(tareas,dibujar);
          }else{
300           dibujar.add(a);
              System.out.println("Tarea agregada: "+a.getNombreTarea());
302           a = manejadorTarea.cargarSubtareas(a);
              if( a.getSubtareas().size()>0){
304             crearListas(a.getSubtareas(),dibujar);
              }
306           crearListas(tareas,dibujar);
          }
308   }
    }
310 public long idProyectoActual(){
      long idProyecto;
312   if((session instanceof SessionMap) && (((SessionMap)session).get("
          idP")!=null)){
        idProyecto = (Long)(((SessionMap)session).get("idP"));
314     return idProyecto;
      }else{
316     return 0;
      }
318 }
    @Override
320 public void setSession(Map<String, Object> arg0) {
      // TODO Auto-generated method stub
322   this.session = arg0;
    }
324 }
```

src/com/gestion/vista/planificacion/TareaAction.java

```
package com.gestion.vista.planificacion;

import java.util.ArrayList;
import java.util.HashSet;
import java.util.Iterator;
import java.util.List;
import java.util.Map;
import java.util.Set;

import org.apache.struts2.dispatcher.SessionMap;
import org.apache.struts2.interceptor.SessionAware;

import com.gestion.modelo.manejadorBd.fase.ManejadorActividad;
import com.gestion.modelo.manejadorBd.fase.ManejadorDisciplina;
import com.gestion.modelo.manejadorBd.fase.ManejadorIteracion;
import com.gestion.modelo.manejadorBd.fase.ManejadorTarea;
import com.gestion.modelo.manejadorBd.proyecto.ManejadorProyecto;
import com.gestion.modelo.manejadorBd.usuario.ManejadorUsuario;
import com.gestion.modelo.proyecto.Actividad;
import com.gestion.modelo.proyecto.Disciplina;
import com.gestion.modelo.proyecto.IdIteracion;
import com.gestion.modelo.proyecto.IdTarea;
import com.gestion.modelo.proyecto.Proyecto;
import com.gestion.modelo.proyecto.Tarea;
import com.gestion.modelo.rol.Rol;
import com.gestion.modelo.usuario.Usuario;
import com.gestion.vista.json.UsuarioJson;
import com.opensymphony.xwork2.ActionSupport;

public class ModificarTareaAction extends ActionSupport implements
    SessionAware{
  private long idProyecto;
  private long numIteracion;
  private long numTarea;
  Map session;
  private Tarea tarea;
  private List<Tarea> padresCandidatos;
  private String padreElegido;

  private List<Tarea> predecesores;
  private List<Tarea> prececesoresCandidatos;
  private List<String> predecesoresElegidos;

  private List<Disciplina> disciplinas;
  private long disciplinaElegida;

  /**
   * Las actividades dependen de la disciplina elegida
   */
  private List<Actividad> actividades;
  private long actividadElegida;
```

```
52  /**
     * Los recursos dependen de la actividad elegida
54   * porque cada actividad esta asociada con un rol
     * solo se cargaran los usuarios que pueden
56   * realizar la actividad
     */
58  private Set<Usuario> recursosActual;
    private List<Usuario> recursosCandidatos;
60  private List<UsuarioJson> recursosCandidatosJson;
    private Set<String> recursosAsignado = new HashSet<String>();
62
    private boolean hito;
64  private boolean pGroup;

66  /**
     * Gestion con la bd
68   *
     */
70  private ManejadorTarea manejadorTarea;
    private ManejadorDisciplina manejadorDisciplina;
72  private ManejadorUsuario manejadorUsuario;
    private ManejadorActividad manejadorActividad;
74  private ManejadorIteracion manejadorIteracion;

76  public ModificarTareaAction(){
      manejadorTarea = new ManejadorTarea();
78    manejadorDisciplina = new ManejadorDisciplina();
      manejadorUsuario = new ManejadorUsuario();
80    manejadorActividad = new ManejadorActividad();
      manejadorIteracion = new ManejadorIteracion();
82  }

84  public List<Tarea> getPadresCandidatos() {
      return padresCandidatos;
86  }

88  public void setPadresCandidatos(List<Tarea> padresCandidatos) {
      this.padresCandidatos = padresCandidatos;
90  }

92  public String getPadreElegido() {
      return padreElegido;
94  }

96  public void setPadreElegido(String padreElegido) {
      this.padreElegido = padreElegido;
98  }

100 public long getIdProyecto() {
      return idProyecto;
102 }
```

```
104   public void setIdProyecto(long idProyecto) {
        this.idProyecto = idProyecto;
106   }

108   public long getNumIteracion() {
        return numIteracion;
110   }

112   public void setNumIteracion(long numIteracion) {
        this.numIteracion = numIteracion;
114   }

116   public long getNumTarea() {
        return numTarea;
118   }

120   public void setNumTarea(long numTarea) {
        this.numTarea = numTarea;
122   }

124

126   public Tarea getTarea() {
        return tarea;
128   }

130   public void setTarea(Tarea tarea) {
        this.tarea = tarea;
132   }

134   public List<Tarea> getPredecesores() {
        return predecesores;
136   }

138   public void setPredecesores(List<Tarea> predecesores) {
        this.predecesores = predecesores;
140   }

142   public List<Tarea> getPrececesoresCandidatos() {
        return prececesoresCandidatos;
144   }

146   public void setPrececesoresCandidatos(List<Tarea>
          prececesoresCandidatos) {
        this.prececesoresCandidatos = prececesoresCandidatos;
148   }

150   public List<String> getPredecesoresElegidos() {
        return predecesoresElegidos;
152   }
```

```
154   public void setPredecesoresElegidos(List<String> predecesoresElegidos)
          {
        this.predecesoresElegidos = predecesoresElegidos;
156   }

158   public List<Disciplina> getDisciplinas() {
        return disciplinas;
160   }

162   public void setDisciplinas(List<Disciplina> disciplinas) {
        this.disciplinas = disciplinas;
164   }

166   public long getDisciplinaElegida() {
        return disciplinaElegida;
168   }

170   public void setDisciplinaElegida(long disciplinaElegida) {
        this.disciplinaElegida = disciplinaElegida;
172   }

174   public List<Actividad> getActividades() {
        return actividades;
176   }

178   public void setActividades(List<Actividad> actividades) {
        this.actividades = actividades;
180   }

182   public long getActividadElegida() {
        return actividadElegida;
184   }

186   public void setActividadElegida(long actividadElegida) {
        this.actividadElegida = actividadElegida;
188   }

190   public Set<Usuario> getRecursosActual() {
        return recursosActual;
192   }

194   public void setRecursosActual(Set<Usuario> recursosActual) {
        this.recursosActual = recursosActual;
196   }

198   public List<Usuario> getRecursosCandidatos() {
        return recursosCandidatos;
200   }

202   public void setRecursosCandidatos(List<Usuario> recursosCandidatos) {
        this.recursosCandidatos = recursosCandidatos;
204   }
```

```
206   public Set<String> getRecursosAsignado() {
        return recursosAsignado;
208   }

210   public void setRecursosAsignado(Set<String> recursosAsignado) {
        this.recursosAsignado = recursosAsignado;
212   }

214   public boolean isHito() {
        return hito;
216   }

218   public void setHito(boolean hito) {
        this.hito = hito;
220   }

222   public boolean ispGroup() {
        return pGroup;
224   }

226   public void setpGroup(boolean pGroup) {
        this.pGroup = pGroup;
228   }

230   public List<UsuarioJson> getRecursosCandidatosJson() {
        return recursosCandidatosJson;
232   }

234   public void setRecursosCandidatosJson(List<UsuarioJson>
         recursosCandidatosJson) {
        this.recursosCandidatosJson = recursosCandidatosJson;
236   }

238   /**
       * Cuando se llama esta accion se le envia tres parametros:
240    * -numTarea
       * -numIteracion
242    * -idProyecto
       * @return
244    */
      public String modificarTarea(){
246     try{
          IdIteracion idIter = new IdIteracion(this.getIdProyecto(),this.
              getNumIteracion());
248       IdTarea id = new IdTarea(this.getIdProyecto(), this.
              getNumIteracion(), this.getNumTarea());

250       tarea = manejadorTarea.datosTarea(id);
          if(tarea != null){
252
            tarea = manejadorTarea.cargarUsuariosTareasPrecedentes(tarea);
```

```
254
        disciplinas = manejadorDisciplina.lista();
256     /**
         * Obtenemos solo las actividades de esta disciplina
258      */

260     actividades = manejadorActividad.lista(tarea.getActividad().
           getDisciplina().getIdDisciplina());
        /**
262      *
         * estrictamente deben ser solo las tareas de este proyecto,
264      * solo los que pertenezcan a la misma disciplina.
         */
266     //padresCandidatos = manejadorIteracion.tareasIteracion(idIter,
           tarea.getActividad().getDisciplina().getIdDisciplina());
        recursosActual = tarea.getRecursos();
268
        /**
270      * Cargamos las tareas del proyecto que pertenezcan a la misma
           disciplina
         */
272     try{
          padresCandidatos = manejadorIteracion.tareasIteracionxxxx(
             idIter, 1);
274       if( padresCandidatos != null){
            padresCandidatos.remove(tarea);
276         System.out.println("$$$$$$$$$$$$$$$$$$44: "+padresCandidatos.
               size());
          }
278     }catch(Exception e){

280     }

282     /**
         * Solo deber ser personal que pertenezca al proyecto y que
           tenga el rol
284      * que especifica la actividad
         */
286     recursosCandidatos = usuariosRol(this.getIdProyecto(),tarea.
           getActividad().getRol().getNombreRol());
        if( tarea.getRecursos().size()>0){
288       Iterator<Usuario> it1 = tarea.getRecursos().iterator();
          while(it1.hasNext()){
290         recursosCandidatos.remove(it1.next());
          }
292     }


294
        predecesores = tarea.getPredecesores();
296
        List<Tarea> aux = manejadorIteracion.tareasIteracion(idIter);
298     Iterator<Tarea> it = aux.iterator();
```

```
        Tarea a;
300     prececesoresCandidatos = new ArrayList<Tarea>();
        while(it.hasNext()){
302       a = it.next();
          if( a.getActividad().getDisciplina().getIdDisciplina()==tarea.
              getActividad().getDisciplina().getIdDisciplina()){
304         if(a.getIdTarea().getNumTarea() != tarea.getIdTarea().
                getNumTarea())
              prececesoresCandidatos.add(a);
306       }
        }
308
      }else{
310     System.out.println("No se pudo cargar la tarea");
      }
312
    }catch(Exception e){
314   e.printStackTrace();
    }
316 return "success";
  }
318
  public String guardarTareaModificada(){
320 try{
    tarea.setIdTarea(new IdTarea(this.getIdProyecto(),this.
        getNumIteracion(),this.getNumTarea()));
322
    /**
324  * Crear la lista del nuevo personal Asignado
     * y asignarlo
326  */

328 Set<Usuario> nuevosRecursos = new HashSet<Usuario>();
    Iterator<String> it = this.getRecursosAsignado().iterator();
330
    String idUser;
332 while(it.hasNext()){
      idUser = it.next();
334   nuevosRecursos.add(manejadorUsuario.buscar(idUser));
    }
336 tarea.setRecursos(nuevosRecursos);

338 /**
     * Cargar la actividad asociada
340  * y asignarla
     */
342 Actividad asociada = manejadorActividad.buscarActividad(
        getActividadElegida());
    tarea.setActividad(asociada);
344
    /**
346  * Cargar Tarea padre
```

```
     * y asignarla
348  */
    Tarea padre = null;
350 if( this.getPadreElegido() == null){
      this.setPadreElegido("");
352 }
    if( this.getPadreElegido().compareTo("") != 0){
354   String vals[];
      vals = this.getPadreElegido().split(" ");
356   IdTarea idPad = new IdTarea(Long.parseLong(vals[0]), Long.
          parseLong(vals[1]), Long.parseLong(vals[2]));
      padre = manejadorTarea.datosTarea(idPad);
358   padre.setpGroup(1);
      padre.setpOpen(1);
360   padre = manejadorTarea.actualizarTarea(padre);
      if(padre.getFechaInicio().after(tarea.getFechaInicio()) ){
362     tarea.setFechaInicio(padre.getFechaInicio());
      }
364 }

366 List<Tarea> padres = new ArrayList<Tarea>();
    padres.add(padre);
368 tarea.setTareaPadre(padres);

370 /**
     * Cargar tareas precedentes
372  */
    List<Tarea> tareaPrecedentes = new ArrayList<Tarea>();
374 Iterator<String> itTareasPrecedentes = predecesoresElegidos.iterator
        ();

376 Tarea predecesor;
    String id[];
378 IdTarea idP;

380 while(itTareasPrecedentes.hasNext()){
      id = itTareasPrecedentes.next().split(" ");
382   idP = new IdTarea(Long.parseLong(id[0]), Long.parseLong(id[1]),
          Long.parseLong(id[2]));
      predecesor = manejadorTarea.datosTarea(idP);
384   if(predecesor == null) continue;

386

      if(predecesor.getFechaFin().after(tarea.getFechaInicio())){
388     tarea.setFechaInicio(predecesor.getFechaFin());
      }
390   tareaPrecedentes.add(predecesor);
    }
392 tarea.setPredecesores(tareaPrecedentes);

394 if( tarea.getFechaInicio().after(tarea.getFechaFin())) tarea.
      setFechaFin(tarea.getFechaInicio());
```

```
396     try{
          tarea = manejadorTarea.actualizarTarea(tarea);
398     }catch(Exception e){
          e.printStackTrace();
400     }
        }catch(Exception e){
402       e.printStackTrace();
        }
404     return "success";
    }
406   public String getJSONListaUsuariosCandidatos(){
        return usuariosToActividad();
408   }
      public String usuariosToActividad(){
410     try{
          /**
412        * El id del proyecto se obtiene de sesion
           */
414       String rolBuscado="";
          Actividad a = manejadorActividad.buscarActividad(this.
              getActividadElegida());
416
          rolBuscado = a.getRol().getNombreRol();
418       /**
           * usuarios del proyecto actual
420        */
          List<Usuario> lista;
422       Long idp = (Long)((SessionMap)session).get("idP");
          if(idp==null){
424         idp=new Long(-1);
          }
426       lista = usuariosRol(idp,rolBuscado);
          Iterator<Usuario> it = lista.iterator();
428
          recursosCandidatosJson = new ArrayList<UsuarioJson>();
430
          while(it.hasNext()){
432         recursosCandidatosJson.add(new UsuarioJson(it.next()));
          }
434     }catch(Exception e){
          e.printStackTrace();
436     }
        return "success";
438   }

440   public List<Usuario> usuariosRol(long idProyecto, String rolBuscado){
        List<Usuario> usuarios;
442     List<Usuario> usrConRol= new ArrayList<Usuario>();
        Iterator<Usuario> it;
444     Iterator<Rol> r;
        Usuario user;
```

```
446     Set<Rol> roles;

448     usuarios = manejadorUsuario.listaUsuarios(idProyecto);
        it= usuarios.iterator();
450

452     while(it.hasNext()){
          user = it.next();
454       user = manejadorUsuario.cargarRoles(user);
          roles = user.getRoles();
456       r = roles.iterator();
          while(r.hasNext()){
458         if(r.next().getNombreRol().compareTo(rolBuscado)==0){
              usrConRol.add(user);
460           break;
            }
462       }
        }
464     return usrConRol;
    }
466
    @Override
468 public void setSession(Map<String, Object> arg0) {
      session = arg0;
470 }

472 }
```

src/com/gestion/vista/planificacion/ModificarTareaAction.java

```
    package com.gestion.vista.planificacion;
2
    import java.util.List;
4



6


8   import com.gestion.modelo.manejadorBd.fase.ManejadorFaseIteracion;
    import com.gestion.modelo.manejadorBd.proyecto.ManejadorProyecto;
10  import com.gestion.modelo.manejadorBd.usuario.ManejadorUsuario;
    import com.gestion.modelo.proyecto.Fase;
12  import com.gestion.modelo.proyecto.Iteracion;
    import com.gestion.modelo.proyecto.NombreFase;
14  import com.gestion.modelo.proyecto.Proyecto;
    import com.gestion.modelo.usuario.Usuario;
16
    public class FaseAction {
18    private List<Fase> fases;
      private List<Iteracion> iteraciones;
20    private Proyecto proyecto;
      private String faseElegida;
22    private int iteracionElegida;
      private ManejadorFaseIteracion manejadorFases;
```

```
24    private ManejadorProyecto manejadorProyecto;
      private ManejadorUsuario manejadorUsuario;
26    public FaseAction() {
        // TODO Auto-generated constructor stub
28      manejadorFases = new ManejadorFaseIteracion();
        manejadorProyecto =  new ManejadorProyecto();
30      manejadorUsuario = new ManejadorUsuario();
      }
32


34

      public List<Fase> getFases() {
36      return fases;
      }
38


40

      public void setFases(List<Fase> fases) {
42      this.fases = fases;
      }
44


46

      public Proyecto getProyecto() {
48      return proyecto;
      }
50


52

      public void setProyecto(Proyecto proyecto) {
54      this.proyecto = proyecto;
      }
56
      public String getFaseElegida() {
58      return faseElegida;
      }
60


62

      public void setFaseElegida(String faseElegida) {
64      this.faseElegida = faseElegida;
      }
66


68

      public int getIteracionElegida() {
70      return iteracionElegida;
      }
72


74

      public void setIteracionElegida(int iteracionElegida) {
```

```
76      this.iteracionElegida = iteracionElegida;
    }
78
80
    public List<Iteracion> getIteraciones() {
82      return iteraciones;
    }
84
86
    public void setIteraciones(List<Iteracion> iteraciones) {
88      this.iteraciones = iteraciones;
    }
90
92
    /**
94   * Obtiene la lista de fases del proyecto especificado con idProyecto
     * es cual es una variable de sesion.
96   * @return
     */
98  public String fases(){
      try{
100     proyecto = manejadorProyecto.iteraciones(4);

102   }catch(Exception e){
        e.printStackTrace();
104   }
      return "success";
106 }

108 /**
     * Obtiene la iteraciones de la fase elegida.
110  * @return
     */
112 public String iteraciones(){
      try{
114     iteraciones = manejadorFases.iteracionesProyecto(4, this.
            getFaseElegida());
      }catch(Exception e){
116     e.printStackTrace();
      }
118   return "success";
    }
120
    public String tareas(){
122   return "success";
    }
124
    public String nuevaIteracion(){
126   return "success";
```

```
  }
}
```

src/com/gestion/vista/planificacion/FaseAction.java

```
package com.interceptores;

import java.util.List;
import java.util.Map;
import java.util.Set;

import com.gestion.modelo.manejadorBd.usuario.ManejadorUsuario;
import com.gestion.modelo.rol.Rol;
import com.gestion.modelo.usuario.Usuario;
import com.opensymphony.xwork2.Action;
import com.opensymphony.xwork2.ActionContext;
import com.opensymphony.xwork2.ActionInvocation;
import com.opensymphony.xwork2.interceptor.Interceptor;

public class ValidarPermisos implements Interceptor{
  ManejadorUsuario manejadorUsuario = new ManejadorUsuario();
  @Override
  public void destroy() {
    // TODO Auto-generated method stub

  }

  @Override
  public void init() {
    // TODO Auto-generated method stub

  }

  @Override
  public String intercept(ActionInvocation action) throws Exception {
    ActionContext actionC = action.getInvocationContext();
    System.out.println("La accion que se desea ejecutar es: "+ actionC.
       getName());
    Map session = action.getInvocationContext().getSession();
     if (session.get("usuario") == null) {
          System.out.println("No esta logueado");
          return Action.LOGIN;
     } else {
          Usuario u = (Usuario)session.get("usuario");
          System.out.println("Usuario logueado: "+u.getCorreo());
          if(validarAccion(u.getRoles(),actionC.getName())){
          return action.invoke();
        }else{
          return "sinpermisos";
        }
        }
  }
```

```
48    public boolean validarAccion(Set<Rol> roles, String accion){
        try{
50        return manejadorUsuario.permitido(roles, accion);
        }catch (Exception e) {
52        // TODO: handle exception
          e.printStackTrace();
54      }
        return false;
56    }

58  }
```

src/com/interceptores/ValidarPermisos.java

Apéndice B

Códigos fuentes de interfaz de usuario

```
<div id="middle">
    <div id="left-column">
      <h3>Header</h3>
      <ul class="nav">
        <li><a href="#">Lorem Ipsum dollar</a></li>
        <li><a href="#">Dollar</a></li>
        <li><a href="#">Lorem dollar</a></li>
        <li><a href="#">Ipsum dollar</a></li>
        <li><a href="#">Lorem Ipsum dollar</a></li>
        <li class="last"><a href="#">Dollar Lorem Ipsum</a></li>
      </ul>
      <a href="#" class="link">Link here</a>
      <a href="#" class="link">Link here</a>
    </div>

    <div id="center-column">
<!-- ################################################################ -->
    Prueba...



<!-- ################################################################ -->
    </div>
    <div id="right-column">
      <strong class="h">INFO</strong>
```

```
28        <div class="box">Detect and eliminate viruses and Trojan horses,
              even new and unknown ones. Detect and eliminate viruses and
              Trojan horses, even new and </div>
      </div>
30 </div>
```

jsp/plantilla/middle.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>
2

4 <div id="footer">
<p>
6   Universidad Autonoma Metropolitana
</p>
8 </div>
```

jsp/plantilla/footer.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>
2   <div id="header2">
      <a href="index.html" class="logo"><img src="img/logo.gif" width="101
          " height="29" alt="" /></a>
4     <ul id="top-navigation">
<!--        <li class="active"><span><span>Inicio</span></span></li>    -->
6       <li class="active"><span><span><a href="<s:url action="inicio"/>">
            Inicio</a></span></span></li>
        <li><span><span><a href="<s:url action="proyectosUsuario"/>">
            Proyectos</a></span></span></li>
8       <li><span><span><a href="<s:url action="personalProyecto"/>">
            Personal</a></span></span></li>
        <li><span><span><a href="<s:url action="iteraciones"/>">Planificar
            </a></span></span></li>
10      <li><span><span><a href="actividadesRup">Actividades</a></span></
            span></li>
        <li><span><span><a href="rolesSistema">Rol</a></span></span></li>
12      <li><span><span><a href="artefactosSistema">Artefactos</a></span
            ></span></li>
        <li><span><span><a href="tareasUsuario">Tareas</a></span></span></
            li>
14    </ul>
    </div>
```

jsp/plantilla/header.jsp

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
      pageEncoding="UTF-8" %>
3 <%@ taglib prefix="s" uri="/struts-tags" %>
<%@ taglib uri="http://tiles.apache.org/tags-tiles" prefix="tiles" %>
5 <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://
    www.w3.org/TR/html4/loose.dtd">
<html>
7 <head>
```

```
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title><tiles:getAsString name="pageTitle"/></title>
    <link href="<s:url value="/jsp/css/admin.css"/>" rel="stylesheet"
          type="text/css"/>
<link type="text/css" href="/PrototipoWeb/jsp/css/jquery-ui-1.8.2.custom
    .css" rel="Stylesheet" />
<script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery-1.4.2.
    min.js"></script>
<script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery-ui
    -1.8.2.custom.min.js"></script>
<script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery.ui.
    datepicker-es.js"></script>
<script type="text/javascript" src="/PrototipoWeb/jsp/js/sistema.js"></
    script>
</head>
<body>

<div id="main">
  <!-- header -->
  <tiles:insertAttribute name="header"/>
  <!-- middle -->
  <tiles:insertAttribute name="middle"/>

  <!-- footer -->
  <tiles:insertAttribute name="footer"/>
</div>

</body>
</html>
```

jsp/plantilla/estructuraGeneral.jsp

```
<script type="text/javascript" src="/PrototipoWeb/jsp/js/datePicker.js"
    ></script>
<%@ taglib prefix="s" uri="/struts-tags" %>
<script type="text/javascript">
<!--
  x = $(document);
  x.ready(inicializarEventos);

  function inicializarEventos(){
    inicializarEventosDatePicker();
  }
//-->
</script>
<div id="middle">
    <div id="left-column">
      <h3>Menu</h3>
      <ul class="nav">
        <li><a href="<s:url action="proyectosUsuario"/>">Mis Proyectos</
            a></li>
        <li><a href="<s:url action="nuevoProyecto"/>">Nuevo</a></li>
```

```
19          <li><a href="<s:url action="modificarEliminarProyecto"/>">
                Modificar/Eliminar</a></li>

21  <!--
            <li><a href="<s:url action="nuevoProyecto"/>">Nuevo</a></li>
23          <li><a href="<s:url action="modificarProyectos"/>">Modificar</a
                ></li>
            <li><a href="<s:url action="proyectosEliminar"/>">Eliminar</a></
                li>
25          <li><a href="<s:url action="proyectosUsuario"/>">Cambiar
                Proyecto Actual</a></li>
            <li><a href="<s:url action="personalProyecto"/>">Participantes
                del Proyecto</a></li>
27          <li class="last"><a href="#">Dollar Lorem Ipsum</a></li>
     -->
29        </ul>
        </div>
31
        <div id="center-column">
33  <!-- ######################################################### -->
        <div class="top-bar">
35          <a href="<s:url action="salirDelSistema"></s:url>" class="button
                ">Salir </a>
            <h1>Modificar Proyecto</h1>
37          <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
                Modificar</a></div>
        </div><br/>
39      <p>
          Seleccione el proyecto que desea modificar:
41      </p>
        <s:form>
43        <s:select name="idProyectoElegido" list="proyectos"
            headerKey = "0" headerValue="Ninguno"
45          listKey="idProyecto" listValue="nombre"></s:select>
          <td colspan="3"><div align="right">
47          <input type="submit"
                  name="enviar1"
49                value="Consultar"
              onclick=this.form.action="datosProyectoModificarEliminar">
51            </div>
          </td>
53          <s:hidden name="proyecto.idProyecto"></s:hidden>
            <s:textfield name="proyecto.nombre" label="Nombre del Proyecto"
                ></s:textfield>
55          <s:textarea name="proyecto.descripcion" label="Descripcion" cols
                ="25" rows="5"></s:textarea>
            <s:textarea name="proyecto.cliente" label="Cliente" cols="25"
                rows="5"></s:textarea>
57          <s:textfield id="fechaInicio" name="proyecto.fechaInicio" label=
                "Fecha de Inicio"></s:textfield>
            <s:textfield id="fechaFin" name="proyecto.fechaFin" label="Fecha
                 de Finalizacion"></s:textfield>
```

```
59
        <tr>
61        <td><p>Jefe del Proyecto</p></td>
          <td colspan="2"><div align="left">
63      <select name="idJefe" label="Jefe del Proyecto">
          <s:iterator value="jefes">
65        <s:if test="idUsuario==proyecto.jefe.idUsuario">
            <option value='<s:property value="idUsuario"></s:property>
              ' selected><s:property value="nombreCompleto"></s:
              property></option>
67        </s:if><s:else>
            <option value='<s:property value="idUsuario"></s:property>
              '><s:property value="nombreCompleto"></s:property></
              option>
69        </s:else>
          </s:iterator>
71      </select>
        </div>
73      </td>
        </tr>
75
       <td colspan="1"><div align="left"></div>
77      <input type="submit"
            name="enviar2"
79          value="Eliminar Proyecto"
         onclick=this.form.action="eliminarProyecto">
81     </td>
        <td colspan="3"><div align="right">
83      <input type="submit"
            name="enviar2"
85          value="Guardar"
         onclick=this.form.action="guardarProyecto">
87       </div>
       </td>
89   </s:form>
     <br></br>
91   <br></br>
     <a href="<s:url action="proyectosUsuario"/>">Regresar</a>
93 <!-- ################################################################ -->
   </div>
95 <div id="right-column">
     <strong class="h">INFO</strong>
97   <div class="box"></div>
   </div>
99 </div>
```

jsp/proyectos/modificarEliminarProyecto.jsp

```
1
<%@ taglib prefix="s" uri="/struts-tags" %>
3

5 <div id="middle">
```

```
    <div id="left-column">
7     <h3>Menu</h3>
      <ul class="nav">
9       <li><a href="<s:url action="proyectosUsuario"/>">Mis Proyectos</
            a></li>
        <li><a href="<s:url action="nuevoProyecto"/>">Nuevo</a></li>
11      <li><a href="<s:url action="modificarEliminarProyecto"/>">
            Modificar/Eliminar</a></li>
<!--
13      <li><a href="<s:url action="nuevoProyecto"/>">Nuevo</a></li>
        <li><a href="<s:url action="modificarProyectos"/>">Modificar</a
            ></li>
15      <li><a href="<s:url action="proyectosEliminar"/>">Eliminar</a></
            li>
        <li><a href="<s:url action="proyectosUsuario"/>">Cambiar
            Proyecto Actual</a></li>
17      <li><a href="<s:url action="personalProyecto"/>">Participantes
            del Proyecto</a></li>
        <li class="last"><a href="#">Dollar Lorem Ipsum</a></li>
19  -->
      </ul>
21    </div>

23    <div id="center-column">
<!-- ################################################################ -->
25    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
27      <h1>Informacion del Proyecto</h1>
        <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
            Proyectos</a></div>
29    </div><br/>
      <p>
31      Informacion del Proyecto
      </p>
33      Nombre del Proyecto: <s:property value="proyecto.nombre"/><br></br
          >
        Descripcion: <s:property value="proyecto.descripcion"/><br></br>
35      Cliente: <s:property value="proyecto.cliente"/><br></br>
        Fecha de Inicio: <s:property value="proyecto.fechaInicio"/><br></
            br>
37      Fecha de Finalizacion: <s:property value="proyecto.fechaFin"/><br
            ></br>
        Jefe del Proyecto: <s:property value="proyecto.jefe.nombreCompleto
            "/><br></br>
39
      <p>Participantes del Proyecto</p>
41    <s:iterator value="proyecto.listaUsuarios" var="it">
        <s:property value="nombreCompleto"/><br></br>
43    </s:iterator>

45    <p>Iteraciones del Proyecto</p>
```

```
    <p>Fase de Inicio</p>
47  <s:iterator value="proyecto.iteraciones" var="it">
      <s:if test="fase.idFase==1">
49      Iteracion:<s:property value="nombre"/><br></br><br></br>
      </s:if>
51  </s:iterator>

53  <p>Fase de Elaboracion</p>
    <s:iterator value="proyecto.iteraciones" var="it">
55    <s:if test="fase.idFase==2">
        Iteracion:<s:property value="nombre"/><br></br><br></br>
57    </s:if>
    </s:iterator>
59
    <p>Fase de Contruccion</p>
61  <s:iterator value="proyecto.iteraciones" var="it">
      <s:if test="fase.idFase==3">
63      Iteracion:<s:property value="nombre"/><br></br><br></br>
      </s:if>
65  </s:iterator>

67  <p>Fase de Transicion</p>
    <s:iterator value="proyecto.iteraciones" var="it">
69    <s:if test="fase.idFase==4">
        Iteracion:<s:property value="nombre"/><br></br><br></br>
71    </s:if>
    </s:iterator>
73
    <a href="<s:url action="proyectosUsuario"/>">Regresar</a>
75<!-- ############################################################ -->
    </div>
77  <div id="right-column">
      <strong class="h">INFO</strong>
79    <div class="box"></div>
    </div>
81</div>
```

jsp/proyectos/detallesProyecto.jsp

```
1
<%@ taglib prefix="s" uri="/struts-tags" %>
3


5<div id="middle">
    <div id="left-column">
7     <h3>Menu</h3>
      <ul class="nav">
9       <li><a href="<s:url action="obtFormProyecto"/>">Nuevo</a></li>
        <li><a href="#">Modificar</a></li>
11      <li><a href="<s:url action="datosProyectos"/>">Eliminar</a></li>
        <li><a href="<s:url action="listarProyectos"/>">Cambiar Proyecto
            Actual</a></li>
13      <li><a href="#">Lorem Ipsum dollar</a></li>
```

```
        <li class="last"><a href="#">Dollar Lorem Ipsum</a></li>
      </ul>
    </div>

    <div id="center-column">
<!-- ######################################################## -->
    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Proyectos</h1>
        <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
            Proyectos</a></div>
    </div><br/>
    <p>
      Seleccione un proyecto:
    </p>
    <s:form action="personalProyecto">
      <s:select name="ptElegido"
        headerKey="-1"
        headerValue=" <--- seleccione un proyecto --> "
        list="listaProyectos"
        listKey="idProyecto" listValue="nombre"
        onchange="this.form.submit()"></s:select>
    </s:form>

    <s:form action="asignarPersonalProyecto">
      <input type="hidden" name="ptElegido">
      <s:optiontransferselect
        name="seleccionIzquierda"
             leftTitle="Personal Disponible"
             rightTitle="Personal del Proyecto"
             list="listaPersonal"
             listKey="idPersonal"
             listValue="nombreCompleto"
             multiple="true"
             headerKey="headerKey"
             headerValue="--- Seleccione el Personal ---"
             size="12"
             emptyOption="false"
             doubleList="personalActual"
             doubleListKey="idPersonal"
             doubleListValue="nombreCompleto"
             doubleName="seleccionDerecha"
             doubleHeaderKey="doubleHeaderKey"
             doubleMultiple="true"
             doubleSize="12"
             allowUpDownOnLeft="false"
             allowUpDownOnRight="false"
             allowAddAllToLeft="false"
             allowAddAllToRight="false"
             allowSelectAll="false"
    />
```

```
    <s:submit value="Aceptar"></s:submit>
65  </s:form>
    Proyecto Actual: <s:property value="proyecto.nombre"/>
67 <!-- ##################################################################### -->
    </div>
69  <div id="right-column">
      <strong class="h">INFO</strong>
71    <div class="box"> </div>
    </div>
73 </div>
```

jsp/proyectos/asignarPersonal.jsp

```
1
<%@ taglib prefix="s" uri="/struts-tags" %>
3

5 <div id="middle">
    <div id="left-column">
7     <h3>Menu</h3>
      <ul class="nav">
9       <li><a href="<s:url action="proyectosUsuario"/>">Mis Proyectos</
            a></li>
        <li><a href="<s:url action="nuevoProyecto"/>">Nuevo</a></li>
11      <li><a href="<s:url action="modificarEliminarProyecto"/>">
            Modificar/Eliminar</a></li>
      </ul>
13    <a href="<s:url action="salirDelSistema"></s:url>" class="link">
          Salir</a>
      <a href="#" class="link">Iniciar Sesion</a>
15  </div>

17  <div id="center-column">
<!-- ############################################################ -->
19  <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
21      <h1>Proyectos</h1>
        <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
            Mis Proyectos</a></div>
23  </div><br/>
    <s:if test="#session.nombreProyecto==null">
25    <div class="errorMessage">
        <p>No ha elegido ningun proyecto sobre el cual trabjar, elija
            uno.</p>
27    </div>
    </s:if><s:else>
29    <h4>El proyecto actual es: <s:property value="#session.
          nombreProyecto"/></h4>
    </s:else>
31
    <div class="cuadro">
33
```

```
35      <s:form action="establecerProyectoActual">
          <s:select name="idProyectoElegido" list="proyectos"
37            listKey="idProyecto" listValue="nombre"></s:select>
          <s:submit value="Aceptar"></s:submit>
39      </s:form>
        </div>
41

43      <p>
        <h3>Descripcion Resumida de los Proyectos</h3>
45      </p>

47      <p>
          <s:iterator value="proyectos" var="it">
49          <h4>Nombre: <s:property value="nombre"/></h4>
            Descripcion: <s:property value="descripcion"/>
51
            <s:url id="detalles" action="detallesProyecto">
53                  <s:param name="idProyectoElegido">
                      <s:property value="idProyecto"/>
55                  </s:param>
                </s:url><br></br>
57          <s:a href="%{detalles}">ver Detalles</s:a>       &
                nbsp

59          <s:url id="modElimP" action="datosProyectoModificarEliminar">
                    <s:param name="idProyectoElegido">
61                    <s:property value="idProyecto"/>
                    </s:param>
63              </s:url>
                <s:a href="%{modElimP}">Modificar/Eliminar</s:a>
65        </s:iterator>
        </p>
67
<!-- ############################################################ -->
69      </div>
        <div id="right-column">
71        <strong class="h">INFO</strong>
          <div class="box"> </div>
73      </div>
</div>
```

jsp/proyectos/proyectosUsuario.jsp

```
2<%@ taglib prefix="s" uri="/struts-tags" %>

4
<div id="middle">
6       <div id="left-column">
          <h3>Menu</h3>
8         <ul class="nav">
```

```
        <li><a href="<s:url action="proyectosUsuario"/>">Mis Proyectos</
            a></li>
10      <li><a href="<s:url action="nuevoProyecto"/>">Nuevo</a></li>
        <li><a href="<s:url action="modificarEliminarProyecto"/>">
            Modificar/Eliminar</a></li>
12    <!--
        <li><a href="<s:url action="nuevoProyecto"/>">Nuevo</a></li>
14      <li><a href="<s:url action="modificarProyectos"/>">Modificar</a
            ></li>
        <li><a href="<s:url action="proyectosEliminar"/>">Eliminar</a></
            li>
16      <li><a href="<s:url action="proyectosUsuario"/>">Cambiar
            Proyecto Actual</a></li>
        <li><a href="<s:url action="personalProyecto"/>">Participantes
            del Proyecto</a></li>
18      <li class="last"><a href="#">Dollar Lorem Ipsum</a></li>
         -->
20    </ul>
    </div>
22
    <div id="center-column">
24<!-- ################################################################## -->
    <div class="top-bar">
26      <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Proyecto Modificado/Eliminado</h1>
28      <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
            Proyectos</a></div>
    </div><br/>
30  <p>

32    <br></br>
    </p>
34
      Nombre del Proyecto: <s:property value="proyecto.nombre"/><br></br
          >
36    Descripcion: <s:property value="proyecto.descripcion"/><br></br>
      Cliente: <s:property value="proyecto.cliente"/><br></br>
38      Fecha de Inicio: <s:property value="proyecto.fechaInicio"/><br
            ></br>
        Fecha de Finalizacion: <s:property value="proyecto.fechaFin"/><
            br></br>
40      Jefe del Proyecto: <s:property value="proyecto.jefe.
            nombreCompleto"/><br><br>
                <s:property value="proyecto.jefe.apellido_paterno"/><
                    br></br>
42              <s:property value="proyecto.jefe.apellido_materno"/><
                    br></br>

44      <a href="<s:url action="modificarEliminarProyecto"/>">Regresar</
            a>
```

```
<!-- ############################################################ -->
    </div>
    <div id="right-column">
      <strong class="h">INFO</strong>
      <div class="box"></div>
    </div>
</div>
```

jsp/proyectos/proyectoModificadoEliminado.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>


<div id="middle">
    <div id="left-column">
      <h3>Menu</h3>
      <ul class="nav">
        <li><a href="<s:url action="nuevoProyecto"/>">Nuevo</a></li>
        <li><a href="<s:url action="modificarProyectos"/>">Modificar</a
            ></li>
        <li><a href="<s:url action="proyectosEliminar"/>">Eliminar</a></
            li>
        <li><a href="<s:url action="proyectosUsuario"/>">Cambiar
            Proyecto Actual</a></li>
        <li><a href="<s:url action="personalProyecto"/>">Participantes
            del Proyecto</a></li>
        <li class="last"><a href="#">Dollar Lorem Ipsum</a></li>
      </ul>
    </div>

    <div id="center-column">
<!-- ############################################################ -->
    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Participantes del Proyecto</h1>
        <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
            Proyectos</a></div>
    </div><br/>
    <p>
    El Proyecto tiene asignado el siguiente personal:
    </p>
    <s:iterator value="personalActual" status="status">
      <s:property value="nombreCompleto"/><br></br>
    </s:iterator>
    <br></br>
    <br></br>
    <a href="<s:url action="personalProyecto"/>">Regresar</a>

<!-- ############################################################ -->
    </div>
    <div id="right-column">
```

```
38        <strong class="h">INFO</strong>
          <div class="box"></div>
40      </div>
</div>
```

jsp/proyectos/personalAsignado.jsp

```
1 <%@ taglib prefix="s" uri="/struts-tags" %>
  <div id="header">
3   <a href="index.html" class="logo"><img src="img/logo.gif" width="101
        " height="29" alt="" /></a>
    Proyecto: <s:property value="#session.nombreProyecto"/><br></br>
5   Usuario: <s:property value="#session.nombreUsuario"/><br></br>
    <ul id="top-navigation">
7     <li><span><span><a href="<s:url action="inicio"/>">Inicio</a></
          span></span></li>
      <li class="active"><span><span><a href="<s:url action="
          proyectosUsuario"/>">Proyecto</a></span></span></li>
9     <li><span><span><a href="<s:url action="personalProyecto"/>">
          Personal</a></span></span></li>
      <li><span><span><a href="<s:url action="iteraciones"/>">Planificar
          </a></span></span></li>
11    <li><span><span><a href="actividadesRup">Actividades</a></span></
          span></li>
      <li><span><span><a href="rolesSistema">Rol</a></span></span></li>
13    <li><span><span><a href="artefactosSistema">Artefactos</a></span
          ></span></li>
      <li><span><span><a href="tareasUsuario">Tareas</a></span></span></
          li>
15    <li><span><span><a href="verPermisos">Seguridad</a></span></span
          ></li>
    </ul>
17  </div>
```

jsp/proyectos/header.jsp

```
1 <%@ taglib prefix="s" uri="/struts-tags" %>

3 <div id="middle">
    <div id="left-column">
5     <h3>Menu</h3>
      <ul class="nav">
7       <li><a href="<s:url action="obtFormProyecto"/>">Nuevo</a></li>
        <li><a href="#">Modificar</a></li>
9       <li><a href="<s:url action="datosProyectos"/>">Eliminar</a></li>
        <li><a href="<s:url action="listarProyectos"/>">Cambiar Proyecto
            Actual</a></li>
11      <li><a href="#">Lorem Ipsum dollar</a></li>
        <li class="last"><a href="#">Dollar Lorem Ipsum</a></li>
13    </ul>
      <a href="#" class="link">Link here</a>
15    <a href="#" class="link">Link here</a>
    </div>
```

```
17
    <div id="center-column">
19 <!-- ############################################################ -->
    <p>
21    Seleccione el proyecto que desea modificar:
    </p>
23  <s:form action="cargarProyectos">
      <s:select name="ptElegido" list="listaP"
25      listKey="idProyecto" listValue="nombre"
        onchange="this.form.submit()"></s:select>
27  </s:form>
    <s:form action="guardarProyecto">
29      <s:textfield name="proyecto.idProyecto" label="Id Proyecto"></s:
            textfield>
        <s:textfield name="proyecto.nombre" label="Nombre del Proyecto"
            ></s:textfield>
31      <s:textarea name="proyecto.descripcion" label="Descripcion" cols
            ="25" rows="5"></s:textarea>
        <s:textarea name="proyecto.cliente" label="Cliente" cols="25"
            rows="5"></s:textarea>
33      <s:textfield name="proyecto.fecha_inicio" label="Fecha de Inicio
            "></s:textfield>
        <s:textfield name="proyecto.fecha_fin" label="Fecha de
            Finalizacion"></s:textfield>
35      <s:textfield name="proyecto.jefe.idPersonal" label="ID Jefe
            Proyecto"></s:textfield>
          <s:submit value="Guardar Proyecto"></s:submit>
37  </s:form>

39 <!-- ############################################################ -->
    </div>
41  <div id="right-column">
      <strong class="h">INFO</strong>
43    <div class="box"></div>
    </div>
45 </div>
```

jsp/proyectos/modificarProyectoOld.jsp

```
1 <script type="text/javascript" src="/PrototipoWeb/jsp/js/datePicker.js"
    ></script>
  <%@ taglib prefix="s" uri="/struts-tags" %>
3 <script type="text/javascript">
  <!--
5   x = $(document);
    x.ready(inicializarEventos);
7
    function inicializarEventos(){
9     inicializarEventosDatePicker();
    }
11 //-->
  </script>
13 <div id="middle">
```

```
    <div id="left-column">
      <h3>Menu</h3>
      <ul class="nav">
        <li><a href="<s:url action="proyectosUsuario"/>">Mis Proyectos</
            a></li>
        <li><a href="<s:url action="nuevoProyecto"/>">Nuevo</a></li>
        <li><a href="<s:url action="modificarEliminarProyecto"/>">
            Modificar/Eliminar</a></li>
<!--
        <li><a href="<s:url action="nuevoProyecto"/>">Nuevo</a></li>
        <li><a href="<s:url action="modificarProyectos"/>">Modificar</a
            ></li>
        <li><a href="<s:url action="proyectosEliminar"/>">Eliminar</a></
            li>
        <li><a href="<s:url action="proyectosUsuario"/>">Cambiar
            Proyecto Actual</a></li>
        <li><a href="<s:url action="personalProyecto"/>">Participantes
            del Proyecto</a></li>
        <li class="last"><a href="#">Dollar Lorem Ipsum</a></li>
 -->
      </ul>
    </div>

    <div id="center-column">
<!-- ################################################################ -->
    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Nuevo Proyecto</h1>
        <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
            Nuevo Proyecto</a></div>
    </div><br/>
      <p>
      Ingrese los datos del nuevo proyecto
      </p>
      <s:form action="registrarProyecto">
        <s:textfield name="proyecto.nombre" label="Nombre del Proyecto"
            ></s:textfield>
        <s:textarea name="proyecto.descripcion" label="Descripcion" cols
            ="25" rows="5"></s:textarea>
        <s:textarea name="proyecto.cliente" label="Cliente" cols="25"
            rows="5"></s:textarea>
        <s:textfield id="fechaInicio" name="proyecto.fechaInicio" label=
            "Fecha de Inicio"></s:textfield>
        <s:textfield id="fechaFin" name="proyecto.fechaFin" label="Fecha
             de Finalizacion"></s:textfield>
        <s:select name="idJefe" list="usuarios" listKey="idUsuario"
            listValue="nombreCompleto" label="Jefe de Proyecto" size="5"
            ></s:select>
        <s:submit value="Aceptar"></s:submit>
      </s:form>
```

```
51
    <a href="<s:url action="proyectosUsuario"/>">Regresar</a>
53
<!-- ################################################################ -->
55    </div>
    <div id="right-column">
57      <strong class="h">INFO</strong>
      <div class="box"></div>
59    </div>
</div>
```

jsp/proyectos/nuevoProyecto.jsp

```
2<%@ taglib prefix="s" uri="/struts-tags" %>

4<div id="middle">
    <div id="left-column">
6     <h3>Menu</h3>
      <ul class="nav">
8       <li><a href="<s:url action="proyectosUsuario"/>">Mis Proyectos</
            a></li>
        <li><a href="<s:url action="nuevoProyecto"/>">Nuevo</a></li>
10      <li><a href="<s:url action="modificarEliminarProyecto"/>">
            Modificar/Eliminar</a></li>

12      <!--
        <li><a href="<s:url action="nuevoProyecto"/>">Nuevo</a></li>
14      <li><a href="<s:url action="modificarProyectos"/>">Modificar</a
            ></li>
        <li><a href="<s:url action="proyectosEliminar"/>">Eliminar</a></
            li>
16      <li><a href="<s:url action="proyectosUsuario"/>">Cambiar
            Proyecto Actual</a></li>
        <li><a href="<s:url action="personalProyecto"/>">Participantes
            del Proyecto</a></li>
18      <li class="last"><a href="#">Dollar Lorem Ipsum</a></li>
         -->
20    </ul>
    </div>
22
    <div id="center-column">
24<!-- ################################################################ -->
    <div class="top-bar">
26      <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Proyecto Registrado</h1>
28      <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
            Proyectos</a></div>
    </div><br/>
30
    <p>
32  El siguiente proyecto se ha creado correctamente.
```

```
    </p>
      Nombre del Proyecto: <s:property value="proyecto.nombre"/><br></br
          >
      Descripcion: <s:property value="proyecto.descripcion"/><br></br>
      Cliente: <s:property value="proyecto.cliente"/><br></br>
      Fecha de Inicio: <s:property value="proyecto.fechaInicio"/><br></
          br>
      Fecha de Finalizacion: <s:property value="proyecto.fechaFin"/><br
          ></br>
      Jefe del Proyecto: <s:property value="proyecto.jefe.nombreCompleto
          "/><br></br>


    <a href="<s:url action="nuevoProyecto"/>">Regresar</a>

<!-- ############################################################ -->
    </div>
    <div id="right-column">
      <strong class="h">INFO</strong>
      <div class="box"> </div>
    </div>
</div>
```

jsp/proyectos/proyectoRegistrado.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>

<div id="middle">
    <div id="left-column">
      <h3>Menu</h3>
      <ul class="nav">

      </ul>
    </div>

    <div id="center-column">
<!-- ############################################################ -->
    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Roles</h1>
        <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
            Proyectos</a></div>
    </div><br/>
      <h3>Rol</h3>
      <s:property value="rol.nombre_rol"></s:property>
      <h3>Descripcion</h3>
      <s:property value="rol.descripcion"></s:property>
      <h3>Categoria</h3>
      <s:property value="rol.categoria"></s:property>
      <h3>Habilidades</h3>
      <s:property value="rol.habilidades"/>
```

```
      <h3>Criterios de Asignacion</h3>
28    <s:property value="rol.criterioAsigacion"/>

30    <a href="javascript:history.back()">Atras</a>
<!-- ############################################################ -->
32  </div>
    <div id="right-column">
34    <strong class="h">INFO</strong>
      <div class="box"></div>
36  </div>
</div>
```

jsp/tareas/verDetallesRol.jsp

```
1
<%@ taglib prefix="s" uri="/struts-tags" %>
3
<div id="middle">
5   <div id="left-column">
      <h3>Menu</h3>
7     <ul class="nav">

9     </ul>
    </div>
11
    <div id="center-column">
13<!-- ############################################################ -->
    <div class="top-bar">
15      <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Roles</h1>
17      <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
            Proyectos</a></div>
    </div><br/>
19  <h3>Subido exitosamente</h3>

21  <a href="javascript:history.back()">Atras</a>
<!-- ############################################################ -->
23  </div>
    <div id="right-column">
25    <strong class="h">INFO</strong>
      <div class="box"> </div>
27  </div>
</div>
```

jsp/tareas/subidoExitosamente.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>
2
<div id="middle">
4   <div id="left-column">
      <h3>Menu</h3>
6     <ul class="nav">
```

```
        <li><a href="<s:url action=""/>">Nueva Iteracion</a></li>
        <li><a href="#">Modificar</a></li>
        <li><a href="<s:url action=""/>">Eliminar</a></li>
        <li><a href="<s:url action=""/>">Cambiar Proyecto Actual</a></li
            >
      </ul>
    </div>

    <div id="center-column">
<!-- ############################################################ -->
    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Artefactos Tarea</h1>
        <div class="breadcrumbs"><a href="#">Tareas</a> / <a href="#">
            Artefactos Tarea</a>
        </div><br/>
    </div>
    <div class="contenidoPrincipal">
      <s:hidden name="numProyecto"></s:hidden>
      <s:hidden name="numIteracion"></s:hidden>
      <s:hidden name="numTarea"></s:hidden>
      <h3>Artefactos Requeridos</h3>
      <s:iterator value="artefactosRequeridos" var="it">
        <s:url id="descargarArtefacto" action="descargarArchivo">
          <s:param name="nombreArtefacto">
            <s:property value="nombre"/>
          </s:param>
          <s:param name="numProyecto">
            <s:property value="numProyecto"/>
          </s:param>
          <s:param name="numIteracion">
            <s:property value="numIteracion"/>
          </s:param>
          <s:param name="numTarea">
            <s:property value="numTarea"/>
          </s:param>
        </s:url>
        <s:url id="descargarPlantilla" action="descargarPlantilla">
          <s:param name="nombreArtefacto">
            <s:property value="nombre"/>
          </s:param>
          <s:param name="numProyecto">
            <s:property value="numProyecto"/>
          </s:param>
          <s:param name="numIteracion">
            <s:property value="numIteracion"/>
          </s:param>
          <s:param name="numTarea">
            <s:property value="numTarea"/>
          </s:param>
        </s:url>
```

```
56          <b><s:property value="nombre"/></b><br></br>
            <s:a href="%{descargarArtefacto}">Descargar Artefacto</s:a>
58          <s:a href="%{descargarPlantilla}">Descargar Plantilla</s:a>
        </s:iterator>
60      <h3>Artefactos Producidos</h3>
        <s:iterator value="artefactosProducidos" var="it">
62        <s:url id="subirArtefacto" action="seleccionarArchivo">
            <s:param name="nombreArtefacto">
64            <s:property value="nombre"/>
            </s:param>
66          <s:param name="numProyecto">
              <s:property value="numProyecto"/>
68          </s:param>
            <s:param name="numIteracion">
70            <s:property value="numIteracion"/>
            </s:param>
72          <s:param name="numTarea">
              <s:property value="numTarea"/>
74          </s:param>
          </s:url>
76        <s:url id="descargarArtefacto2" action="descargarArchivo">
            <s:param name="nombreArtefacto">
78            <s:property value="nombre"/>
            </s:param>
80          <s:param name="numProyecto">
              <s:property value="numProyecto"/>
82          </s:param>
            <s:param name="numIteracion">
84            <s:property value="numIteracion"/>
            </s:param>
86          <s:param name="numTarea">
              <s:property value="numTarea"/>
88          </s:param>
          </s:url>
90
          <b><s:property value="nombre"/></b><br></br>
92        <s:a href="%{subirArtefacto}">Subir Artefacto</s:a>
          <s:a href="%{descargarArtefacto2}">Descargar Artefacto</s:a>
94      </s:iterator>

96      <br></br><br></br>
        <a href="tareasUsuario">Regresar</a>
98    </div>
<!-- ############################################################## -->
100   </div>
      <div id="right-column">
102     <strong class="h">INFO</strong>
        <div class="box"></div>
104   </div>
</div>
```

jsp/tareas/artefactosTarea.jsp

```
1
<%@ taglib prefix="s" uri="/struts-tags" %>
3<div id="middle">
    <div id="left-column">
5     <h3>Menu</h3>
      <ul class="nav">
7
      </ul>
9   </div>

11  <div id="center-column">
<!-- ############################################################ -->
13  <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
15      <h1>Actividad: <s:property value="actividad.nombre"></s:property
            ></h1>
        <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
            Proyectos</a></div>
17  </div><br/>

19  <h3>Actividad</h3>
    <s:property value="actividad.nombre"></s:property>
21
    <h3>Descripcion</h3>
23  <s:property value="actividad.descripcion"/>

25  <h3>Rol</h3>
    <s:property value="actividad.ron.nombreRol"/>
27
    <h3>Disciplina</h3>
29  <s:property value="actividad.disciplina.nombre"/>

31  <h3>Informacion General</h3>
    <s:property value="actividad.informacionGeneral"/>
33
    <h3>Artefactos Requeridos</h3>
35  <s:iterator value="actividad.requeridos" var="it">
      <s:property value="nombre"/><br></br>
37  </s:iterator>

39  <h3>Artefactos Producidos</h3>
    <s:iterator value="actividad.producidos" var="it">
41    <s:property value="nombre"/><br></br>
    </s:iterator>
43
    <a href="javascript:history.back()">Atras</a>
45

47<!-- ############################################################ -->
    </div>
```

```
49      <div id="right-column">
          <strong class="h">INFO</strong>
51        <div class="box"></div>
        </div>
53  </div>
```

jsp/tareas/verDetallesActividad.jsp

```
1
  <%@ taglib prefix="s" uri="/struts-tags" %>
3 <div id="middle">
      <div id="left-column">
5       <h3>Menu</h3>
        <ul class="nav">
7
        </ul>
9     </div>

11    <div id="center-column">
  <!-- ############################################################ -->
13    <div class="top-bar">
          <a href="<s:url action="salirDelSistema"></s:url>" class="button
              ">Salir </a>
15        <h1>Tareas</h1>
          <div class="breadcrumbs"><a href="#">Tareas</a> / <a href="#">
              Tareas</a></div>
17    </div><br/>

19    <table class="tareas">
        <caption>Tareas Asignadas</caption>
21      <thead>
          <tr>
23          <th>Proyecto</th>
            <th>Fase</th>
25          <th>Iteracion</th>
            <th>Tarea</th>
27          <th>Artefactos</th>
          </tr>
29      </thead>
        <tbody>
31        <s:iterator value="proyectosUsuario" var="it">
          <s:iterator value="usuario.tareas" var="it2">
33          <s:if test="idProyecto == idTarea.idProyecto">
            <s:url id="url" action="detallesTareaUsuario">
35            <s:param name="numProyecto">
                <s:property value="idTarea.idProyecto"/>
37            </s:param>
              <s:param name="numIteracion">
39              <s:property value="idTarea.numIteracion"/>
              </s:param>
41            <s:param name="numTarea">
                <s:property value="idTarea.numTarea"/>
43            </s:param>
```

```
            </s:url>
            <s:url id="artefactosTarea" action="artefactosTarea">
              <s:param name="numProyecto">
                <s:property value="idTarea.idProyecto"/>
              </s:param>
              <s:param name="numIteracion">
                <s:property value="idTarea.numIteracion"/>
              </s:param>
              <s:param name="numTarea">
                <s:property value="idTarea.numTarea"/>
              </s:param>
            </s:url>
            <tr>
              <td><s:property value="nombre"/></td>
              <td><s:property value="iteracion.fase.nombre"/></td>
              <td><s:property value="iteracion.nombre"/></td>
              <td><s:property value="nombreTarea"/></td>
              <td><s:a href="%{artefactosTarea}">Artefactos</s:a></td>
              <td> <s:a href="%{url}">Detalles</s:a> </td>
            </tr>
            </s:if>
          </s:iterator>
          </s:iterator>

        </tbody>
      </table>
<!-- ############################################################ -->
      </div>
      <div id="right-column">
        <strong class="h">INFO</strong>
        <div class="box"></div>
      </div>
</div>
```

jsp/tareas/tareasUsuario.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>

<div id="middle">
      <div id="left-column">
        <h3>Menu</h3>
        <ul class="nav">
          <li><a href="<s:url action=""/>">Nueva Iteracion</a></li>
          <li><a href="#">Modificar</a></li>
          <li><a href="<s:url action=""/>">Eliminar</a></li>
          <li><a href="<s:url action=""/>">Cambiar Proyecto Actual</a></li
              >
        </ul>
      </div>

      <div id="center-column">
<!-- ############################################################ -->
      <div class="top-bar">
```

```
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
18      <h1>Detalles de la Tarea</h1>
        <div class="breadcrumbs"><a href="#">Tareas</a> / <a href="#">
            Detalles Tarea</a>
20      </div><br/>
    </div>
22  <div class="contenidoPrincipal">

24      <b>Tarea: </b>
        <s:property value="tarea.nombreTarea"/><br></br>
26
        <b>Fecha de Inicio: </b>
28      <s:property value="tarea.fechaInicio"/><br></br>

30      <b>Fecha de Terminacion:</b>
        <s:property value="tarea.fechaFin"/><br></br>
32
        <b>Porcentaje Completado:</b>
34      <s:property value="tarea.porcentaje"/><br></br>

36      <b>Disciplina:</b>
        <s:property value="tarea.actividad.disciplina.nombre"/><br></br>
38
        <b>Actividad:</b>
40      <s:url id="urlActividad" action="verDetallesActividad">
          <s:param name="idActividad">
42          <s:property value="tarea.actividad.idActividad"/>
          </s:param>
44      </s:url>
        <s:a href="%{urlActividad}">
46        <s:property value="tarea.actividad.nombre"/>
        </s:a><br></br>
48


50      <b>Rol:</b>
        <s:url id="urlRol" action="verDetallesRol">
52        <s:param name="nombreRol">
            <s:property value="tarea.actividad.rol.nombreRol"/>
54        </s:param>
        </s:url>
56      <s:a href="%{urlRol}"><s:property value="tarea.actividad.rol.
            nombreRol"/></s:a><br></br>

58      <br></br><br></br>
        <a href="tareasUsuario">Regresar</a>
60    </div>
<!-- ############################################################ -->
62  </div>
    <div id="right-column">
64    <strong class="h">INFO</strong>
      <div class="box"> </div>
```

```
66      </div>
</div>
```

jsp/tareas/detallesTareaUsuario.jsp

```
1 <%@ taglib prefix="s" uri="/struts-tags" %>
  <div id="header2">
3     <a href="index.html" class="logo"><img src="img/logo.gif" width="101
          " height="29" alt="" /></a>
      <ul id="top-navigation">
5 <!--        <li class="active"><span><span>Inicio</span></span></li>   -->
        <li><span><span><a href="<s:url action="inicio"/>">Inicio</a></
            span></span></li>
7       <li><span><span><a href="<s:url action="proyectosUsuario"/>">
            Proyectos</a></span></span></li>
        <li><span><span><a href="<s:url action="personalProyecto"/>">
            Personal</a></span></span></li>
9       <li><span><span><a href="<s:url action="iteraciones"/>">Planificar
            </a></span></span></li>
        <li><span><span><a href="actividadesRup">Actividades</a></span></
            span></li>
11      <li><span><span><a href="rolesSistema">Rol</a></span></span></li>
        <li><span><span><a href="artefactosSistema">Artefactos</a></span
            ></span></li>
13      <li class="active"><span><span><a href="tareasUsuario">Tareas</a
            ></span></span></li>
        <li><span><span><a href="verPermisos">Seguridad</a></span></span
            ></li>
15    </ul>
  </div>
```

jsp/tareas/header.jsp

```
  <%@ taglib prefix="s" uri="/struts-tags" %>
2
  <div id="middle">
4     <div id="left-column">
        <h3>Menu</h3>
6       <ul class="nav">
          <li><a href="<s:url action=""/>">Subir/Descargar Artefacto</a></
              li>
8       </ul>
      </div>
10
      <div id="center-column">
12 <!-- ################################################################## -->
      <div class="top-bar">
14        <a href="<s:url action="salirDelSistema"></s:url>" class="button
              ">Salir </a>
          <h1>Detalles de la Tarea</h1>
16        <div class="breadcrumbs"><a href="#">Tareas</a> / <a href="#">
              Detalles Tarea</a>
          </div><br/>
```

```
18      </div>
        <div class="contenidoPrincipal">
20
            <h3>Elija una tarea</h3>
22          <s:select list="tareas" listKey="idTarea.idCadena" listValue="
                nombreTarea">
            </s:select>
24
          </div>
26  <!-- ############################################################ -->
        </div>
28      <div id="right-column">
          <strong class="h">INFO</strong>
30        <div class="box"></div>
        </div>
32  </div>
```

jsp/tareas/entregarDescargarArtefacto.jsp

```
  <%@ taglib prefix="s" uri="/struts-tags" %>
2
  <div id="middle">
4       <div id="left-column">
          <h3>Menu</h3>
6         <ul class="nav">
          </ul>
8       </div>

10      <div id="center-column">
  <!-- ############################################################ -->
12      <div class="top-bar">
            <a href="<s:url action="salirDelSistema"></s:url>" class="button
                ">Salir </a>
14          <h1>Artefactos Tarea</h1>
            <div class="breadcrumbs"><a href="#">Tareas</a> / <a href="#">
                Artefactos Tarea</a>
16          </div><br/>
        </div>
18      <div class="contenidoPrincipal">
          <s:form action="subirArchivoArtefacto" enctype="multipart/form-
              data">
20          <s:hidden name="numProyecto"></s:hidden>
            <s:hidden name="numIteracion"></s:hidden>
22          <s:hidden name="numTarea"></s:hidden>
            <s:hidden name="nombreArtefacto"></s:hidden>
24            <s:file name="attachment" label="Ejemplar del Archivo"/>
              <s:submit value="Aceptar"></s:submit>
26        </s:form>
        </div>
28  <!-- ############################################################ -->
        </div>
30      <div id="right-column">
          <strong class="h">INFO</strong>
```

```
32        <div class="box"></div>
      </div>
34 </div>
```

jsp/tareas/subirArtefacto.jsp

```
2 <%@ taglib prefix="s" uri="/struts-tags" %>
  <div id="middle">
4     <div id="left-column">
        <h3>Menu</h3>
6       <ul class="nav">
          <li class=""><a href="<s:url action="artefactosSistema"/>">
              Artefactos (RUP)</a></li>
8         <li class=""><a href="#">Artefactos del Proyecto</a></li>
          <li class=""><a href="<s:url action="nuevoArtefacto"/>">Nuevo
              Artefacto</a></li>
10        <li class=""><a href="<s:url action="modificarEliminarArtefacto"
              />">Modificar/Eliminar Artefacto</a></li>
        </ul>
12    </div>

14    <div id="center-column">
  <!-- ################################################################ -->
16    <div class="top-bar">
          <a href="<s:url action="salirDelSistema"></s:url>" class="button
              ">Salir </a>
18        <h1>Informacion del Artefacto</h1>
          <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
              Proyectos</a></div>
20    </div><br/>

22    <h3>Artefacto</h3>
      <s:property value="artefacto.nombre"/><br></br>
24
      <h3>Rol</h3>
26    <s:property value="artefacto.rol.nombreRol"/><br></br>

28    <h3>Descripcion</h3>
      <s:property value="artefacto.descripcion" escape="false"/><br></br>
30


32    <h3>Representacion UML</h3>
      <s:property value="artefacto.representacionUML" escape="false"/><br
          ></br>
34
      <h3>Proposito</h3>
36    <s:property value="artefacto.proposito" escape="false"/><br></br>

38    <h3>Momento de Creacion</h3>
      <s:property value="artefacto.timing" escape="false"/><br></br>
40
      <h3>Adaptacion</h3>
```

```
42      <s:property value="artefacto.tailoring" escape="false"/><br></br>

44      <br></br>
        <br></br>
46      <a href="<s:url action=""/>">Regresar</a>

48  <!-- ############################################################ -->
        </div>
50      <div id="right-column">
          <strong class="h">INFO</strong>
52        <div class="box"> </div>
        </div>
54  </div>
```

jsp/tareas/verDetallesArtefacto.jsp

```
2   <%@ taglib prefix="s" uri="/struts-tags" %>
    <div id="middle">
4       <div id="left-column">
          <h3>Menu</h3>
6         <ul class="nav">
            <li class=""><a href="<s:url action="rolesSistema"/>">Roles del
                Sistema</a></li>
8           <li class=""><a href="#">Rol (Proyecto)</a></li>
            <li class=""><a href="<s:url action="nuevoRol"/>">Nuevo Rol</a
                ></li>
10
          </ul>
12      </div>

14      <div id="center-column">
    <!-- ############################################################ -->
16      <div class="top-bar">
            <a href="<s:url action="salirDelSistema"></s:url>" class="button
                ">Salir </a>
18          <h1>Informacion del Rol</h1>
            <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
                Proyectos</a></div>
20      </div><br/>

22      Nombre: <s:property value="rol.nombreRol"/><br></br>

24      <a href="<s:url action="rolesSistema"/>">Regresar</a>

26  <!-- ############################################################ -->
        </div>
28      <div id="right-column">
          <strong class="h">INFO</strong>
30        <div class="box"></div>
        </div>
32  </div>
```

jsp/rol/infoRol.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>
<div id="middle">
    <div id="left-column">
      <h3>Menu</h3>
      <ul class="nav">
        <li class=""><a href="<s:url action="rolesSistema"/>">Rol (RUP)
            </a></li>
        <li class=""><a href="#">Rol (Proyecto)</a></li>
        <li class=""><a href="<s:url action="nuevoRol"/>">Nuevo Rol</a
            ></li>

      </ul>
    </div>

    <div id="center-column">
<!-- ##################################################################### -->
    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Roles</h1>
        <div class="breadcrumbs"><a href="#">Rol</a> / <a href="#">Rol (
            RUP)</a></div>
    </div><br/>

    <p>Roles de RUP</p>

    <s:iterator value="roles" var="it">

      <!-- Agregamos una liga para que el usuario
        vea la informacion detallada del rol -->
      <s:url id="dt1" action="detallesRol2">
          <s:param name="nombreRol">
            <s:property value="nombreRol"/>
          </s:param>
      </s:url>

      <!-- Agregamos una liga para que el usuario
        pueda modificar el artefacto -->
      <s:url id="dt2" action="modificarEliminarRol">
          <s:param name="nombreRol">
            <s:property value="nombreRol"/>
          </s:param>
      </s:url>

      <s:if test="categoria == 'Programador'">
        <h4><s:a href="%{dt1}"><s:property value="nombreRol"/></s:a></h4
            >
        <s:a href="%{dt2}">Modificar/Eliminar</s:a><br></br>
```

```
      </s:if><s:else>
46      <h4><s:a href="%{dt1}"><s:property value="nombreRol"/></s:a></h4
            >
        <s:a href="%{dt2}">Modificar/Eliminar</s:a><br></br>
48    </s:else>
      <br></br>
50  </s:iterator>

52<!-- ############################################################ -->
    </div>
54  <div id="right-column">
      <strong class="h">INFO</strong>
56    <div class="box"></div>
    </div>
58</div>
```

jsp/rol/rolesSistema.jsp

```
 2<%@ taglib prefix="s" uri="/struts-tags" %>
 <div id="middle">
 4    <div id="left-column">
        <h3>Menu</h3>
 6      <ul class="nav">
          <li class=""><a href="<s:url action="rolesSistema"/>">Roles del
              Sistema</a></li>
 8        <li class=""><a href="#">Rol (Proyecto)</a></li>
          <li class=""><a href="<s:url action="nuevoRol"/>">Nuevo Rol</a
              ></li>
10
        </ul>
12    </div>

14    <div id="center-column">
 <!-- ############################################################ -->
16    <div class="top-bar">
          <a href="<s:url action="salirDelSistema"></s:url>" class="button
              ">Salir </a>
18        <h1>Roles</h1>
          <div class="breadcrumbs"><a href="#">Rol</a> / <a href="#">
              Modificar/Eliminar Rol</a></div>
20    </div><br/>

22    <p>Introduzca los datos</p>

24      <s:form>
          <s:textfield name="rol.nombreRol" label="Nombre del Rol"></s:
              textfield>
26        <s:textarea name="rol.descripcion" label="Descripcion"></s:
              textarea>
          <s:textfield name="rol.urlDescripcion" label="Url descripcion"
              ></s:textfield>
```

```
28          <s:select name="rol.categoria" list="{'Administrador','
                Programador','Soporte'}" label="Categoria"></s:select>
            <s:textarea name="rol.habilidades" label="Habilidades"></s:
                textarea>
30          <s:textarea name="rol.criterioAsignacion" label="Criterios
                Asignacion"></s:textarea>

32          <td colspan="1"><div align="left"></div>
              <input type="submit"
34                 name="enviar2"
                   value="Eliminar"
36            onclick=this.form.action="eliminarRol">
            </td>
38          <td colspan="3"><div align="right">
            <input type="submit"
40                 name="enviar2"
                   value="Guardar"
42            onclick=this.form.action="guardarCambiosRol">
              </div>
44          </td>
          </s:form >
46
      <a href="<s:url action="rolesSistema"/>">Regresar</a>
48 <!-- ############################################################ -->
      </div>
50    <div id="right-column">
        <strong class="h">INFO</strong>
52      <div class="box"></div>
      </div>
54 </div>
```

jsp/rol/modificarEliminarRol.jsp

```
2 <%@ taglib prefix="s" uri="/struts-tags" %>

4 <div id="middle">
      <div id="left-column">
6       <h3>Header</h3>
        <ul class="nav">
8         <li><a href="#">Lorem Ipsum dollar</a></li>
          <li><a href="#">Dollar</a></li>
10        <li><a href="#">Lorem dollar</a></li>
          <li><a href="#">Ipsum dollar</a></li>
12        <li><a href="#">Lorem Ipsum dollar</a></li>
          <li class="last"><a href="#">Dollar Lorem Ipsum</a></li>
14      </ul>
      </div>
16 <!-- ############################################################ -->
      <div id="center-column">
18      <p>
        Introduzca los datos del Rol.
20      </p>
```

```
      <s:form action="registrarRol">
22      <s:textfield name="rol.nombre_rol" label="Nombre del Rol"></s:
            textfield>
        <s:textarea name="rol.descripcion" label="Descripcion"></s:
            textarea>
24      <s:textfield name="rol.urlDescripcion" label="Url descripcion"
            ></s:textfield>
        <s:select name="rol.categoria" list="{'Administrador','
            Programador','Soporte'}" label="Categoria"></s:select>
26      <s:submit value="Registrar"></s:submit>
      </s:form >
28
    </div>
30<!-- ############################################################ -->
    <div id="right-column">
32    <strong class="h">INFO</strong>
      <div class="box"></div>
34  </div>
</div>
```

jsp/rol/registrarRol.jsp

```
1
<%@ taglib prefix="s" uri="/struts-tags" %>
3
<div id="middle">
5   <div id="left-column">
      <h3>Header</h3>
7     <ul class="nav">
        <li><a href="#">Lorem Ipsum dollar</a></li>
9       <li><a href="#">Dollar</a></li>
        <li><a href="#">Lorem dollar</a></li>
11      <li><a href="#">Ipsum dollar</a></li>
        <li><a href="#">Lorem Ipsum dollar</a></li>
13      <li class="last"><a href="#">Dollar Lorem Ipsum</a></li>
      </ul>
15    <a href="#" class="link">Link here</a>
      <a href="#" class="link">Link here</a>
17  </div>

19  <div id="center-column">
<!-- ############################################################ -->
21    Prueba...
      <p>
23    Rol Registrado:
      </p>
25    <s:property value="rol.nombre_rol"></s:property>
      <s:property value="rol.descripcion"></s:property>
27    <s:property value="rol.urlDescripcion"></s:property>
      <s:property value="rol.categoria"></s:property>
29
      <a href="<s:url action="nuevoRol"/>">Regresar</a>
31<!-- ############################################################ -->
```

```
    </div>
33  <div id="right-column">
      <strong class="h">INFO</strong>
35    <div class="box"></div>
    </div>
37 </div>
```

jsp/rol/rolAgregado.jsp

```
1 <%@ taglib prefix="s" uri="/struts-tags" %>
  <div id="header">
3   <a href="index.html" class="logo"><img src="img/logo.gif" width="101
        " height="29" alt="" /></a>
    Proyecto: <s:property value="#session.nombreProyecto"/><br></br>
5   Usuario: <s:property value="#session.nombreUsuario"/><br></br>
    <ul id="top-navigation">
7     <li><span><span><a href="<s:url action="inicio"/>">Inicio</a></
          span></span></li>
      <li ><span><span><a href="<s:url action="proyectosUsuario"/>">
          Proyecto</a></span></span></li>
9     <li><span><span><a href="<s:url action="personalProyecto"/>">
          Personal</a></span></span></li>
      <li><span><span><a href="<s:url action="iteraciones"/>">Planificar
          </a></span></span></li>
11    <li><span><span><a href="actividadesRup">Actividades</a></span></
          span></li>
      <li class="active"><span><span><a href="rolesSistema">Rol</a></
          span></span></li>
13    <li><span><span><a href="artefactosSistema">Artefactos</a></span
          ></span></li>
      <li><span><span><a href="tareasUsuario">Tareas</a></span></span></
          li>
15    <li><span><span><a href="verPermisos">Seguridad</a></span></span
          ></li>
    </ul>
17 </div>
```

jsp/rol/header.jsp

```
1
<%@ taglib prefix="s" uri="/struts-tags" %>
3
<div id="middle">
5   <div id="left-column">
      <h3>Header</h3>
7     <ul class="nav">
        <li><a href="#">Lorem Ipsum dollar</a></li>
9       <li><a href="#">Dollar</a></li>
        <li><a href="#">Lorem dollar</a></li>
11      <li><a href="#">Ipsum dollar</a></li>
        <li><a href="#">Lorem Ipsum dollar</a></li>
13      <li class="last"><a href="#">Dollar Lorem Ipsum</a></li>
      </ul>
```

```
15        <a href="#" class="link">Link here</a>
          <a href="#" class="link">Link here</a>
17      </div>
<!-- ############################################################ -->
19      <div id="center-column">

21      Prueba...
          <p>
23        Introduzca los datos del Rol a buscar
          </p>
25        <s:form action="buscarRol">
            <s:textfield name="nombre_rol" label="Nombre del Rol"></s:
                textfield>
27          <s:submit value="Buscar"></s:submit>
          </s:form >
29
          Los datos del rol son los siguientes:
31        Rol: <s:property value="rol.nombre_rol"/><br></br>
          Descripcion: <s:property value="rol.descripcion"/><br></br>
33        Categoria: <s:property value="rol.categoria"/><br></br>


35

      </div>
37<!-- ############################################################ -->
      <div id="right-column">
39      <strong class="h">INFO</strong>
        <div class="box"></div>
41    </div>
</div>
```

jsp/rol/buscarRol.jsp

```

2<%@ taglib prefix="s" uri="/struts-tags" %>
<div id="middle">
4     <div id="left-column">
        <h3>Menu</h3>
6       <ul class="nav">
          <li class=""><a href="<s:url action="rolesSistema"/>">Roles del
              Sistema</a></li>
8         <li class=""><a href="#">Rol (Proyecto)</a></li>
          <li class=""><a href="<s:url action="nuevoRol"/>">Nuevo Rol</a
              ></li>
10


12      </ul>
      </div>
14
      <div id="center-column">
16<!-- ############################################################ -->
      <div class="top-bar">
18        <a href="<s:url action="salirDelSistema"></s:url>" class="button
              ">Salir </a>
```

```
        <h1>Roles</h1>
        <div class="breadcrumbs"><a href="#">Rol</a> / <a href="#">Nuevo
            Rol</a></div>
    </div><br/>

    <p>Introduzca los datos</p>

      <s:form action="guardarRol">
        <s:textfield name="rol.nombreRol" label="Nombre del Rol"></s:
            textfield>
        <s:textarea name="rol.descripcion" label="Descripcion"></s:
            textarea>
        <s:textfield name="rol.urlDescripcion" label="Url descripcion"
            ></s:textfield>
        <s:select name="rol.categoria" list="{'Administrador','
            Programador','Soporte'}" label="Categoria"></s:select>
        <s:textarea name="rol.habilidades" label="Habilidades"></s:
            textarea>
        <s:textarea name="rol.criterioAsignacion" label="Criterios
            Asignacion"></s:textarea>
        <s:submit value="Registrar"></s:submit>
      </s:form >

    <a href="<s:url action="rolesSistema"/>">Regresar</a>
<!-- ################################################################ -->
    </div>
    <div id="right-column">
      <strong class="h">INFO</strong>
      <div class="box"> </div>
    </div>
</div>
```

jsp/rol/nuevoRol.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>
<link type="text/css" href="/PrototipoWeb/jsp/css/jquery-ui-1.8.2.custom
    .css" rel="Stylesheet"/>
<script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery.
    wymeditor.min.js"></script>
<script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery-ui
    -1.8.2.custom.min.js"></script>
<script type="text/javascript">

/* Here we replace each element with class 'wymeditor'
 * (typically textareas) by a WYMeditor instance.
 *
 * We could use the 'html' option, to initialize the editor's content.
 * If this option isn't set, the content is retrieved from
 * the element being replaced.
 */

 var x = $(document);
```

```
 x.ready(inicializarEventos);
18
 function inicializarEventos(){
20  var x = $(".wymeditor");
  x.wymeditor();
22
  x = $("#mostrarDescripcion");
24  x.click(mostrar);

26  x = $("#ocultarDescripcion");
  x.click(ocultar);
28
  var x = $("#acordeonActividad");
30  x.accordion({ autoHeight: false });
 }
32
 function ocultar(){
34  var x = $("#descripcion");
  x.css("display","none");
36  x = $("#text1");
  x.css("display","block");
38 }

40 function mostrar(){
  var x = $("#descripcion");
42  x.css("display","block");
  x = $("#text1");
44  x.css("display","block");
 }
46
</script>
48
<div id="middle">
50    <div id="left-column">
      <h3>Menu</h3>
52      <ul class="nav">
        <li class=""><a href="<s:url action="actividadesRup"/>">
            Actividades (RUP)</a></li>
54      <li class=""><a href="#">Actividades del Proyecto</a></li>
        <li class=""><a href="<s:url action="nuevaActividad"/>">Nuevo
            Actividad</a></li>
56      </ul>
      <a href="<s:url action="salirDelSistema"></s:url>" class="link">
         Salir</a>
58    </div>

60    <div id="center-column">
<!-- ################################################################ -->
62    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
64      <h1>Modificar/Eliminar Actividad</h1>
```

```
        <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
            Proyectos</a></div>
66    </div><br></br>

68    <p>Actividad</p>

70    <form>
      <s:hidden name="actividad.idActividad"></s:hidden>
72
      <s:textfield label="Nombre" name="actividad.nombre"></s:textfield>
74
      <tr>
76    <td><p>Rol</p></td>
      <td colspan="1"><div align="left">
78    <select name="nombreRol">
        <s:iterator value="roles" var="it">
80        <s:if test="actividad.rol.nombreRol == nombreRol">
            <option value='<s:property value="nombreRol"></s:property>'
                selected><s:property value="nombreRol"></s:property></
                option>
82        </s:if><s:else>
            <option value='<s:property value="nombreRol"></s:property>'><s
                :property value="nombreRol"></s:property></option>
84        </s:else>
        </s:iterator>
86    </select>
      </div>
88    </td>
      </tr>
90

92    <tr>
      <td><p>Disciplina</p></td>
94    <td colspan="1"><div align="left">
      <select name="idDisciplina">
96      <s:iterator value="disciplinas" var="it">
          <s:if test="actividad.disciplina.idDisciplina == idDisciplina">
98          <option value='<s:property value="idDisciplina"></s:property>'
                selected><s:property value="nombre"></s:property></option
                >
          </s:if><s:else>
100         <option value='<s:property value="idDisciplina"></s:property>'
                ><s:property value="nombre"></s:property></option>
          </s:else>
102     </s:iterator>
      </select>
104   </div>
      </td>
106   </tr>

108
```

```
110     <br></br>
        <br></br>
112     <s:div id="acordeonActividad">
          <h3><a href="#">Proposito</a></h3>
114       <s:div>
            <tr>
116         <td colspan="2"><div id="descripcion" align="left">
            <textarea id="text1" class="wymeditor" name="actividad.
                descripcion">
118           <s:property value="actividad.descripcion"/>
            </textarea>
120         <input type="button" value="Aceptar" class="wymupdate" />
            </div>
122         </td>
            </tr>
124       </s:div>

126       <h3><a href="#">Artefactos de Entrada</a></h3>
          <s:div>
128         <p>
            Aqui ponemos informacion de artefactos de entrada.
130         </p>

132         <s:optiontransferselect
                    id="artefactosEntrada"
134                 name="artefactosEntrada"
                        leftTitle="Artefactos Requeridoss"
136                     rightTitle="Artefactos Disponibles"
                        list="actividad.requeridos"
138                     listKey="nombre"
                        listValue="nombre"
140                     multiple="true"
                        headerKey="0"
142                     headerValue="---------- Artefactos Requeridos
                            ----------"
                        size="6"
144                     doubleId="artefactosDisponibles1"
                        doubleList="artefactos"
146                     doubleListKey="nombre"
                        doubleListValue="nombre"
148                     doubleName=""
                        doubleHeaderKey="0"
150                     doubleHeaderValue="<---------- Artefactos Disponibles
                            ---------->"
                        doubleMultiple="true"
152                     doubleSize="6"
                        allowUpDownOnLeft="false"
154                     allowUpDownOnRight="false"
                        allowAddAllToLeft="false"
156                     allowAddAllToRight="false"
                        allowSelectAll="false"
158               />
```

```
160
162         </s:div>
164         <h3><a href="#">Artefactos de Salida</a></h3>
            <s:div>
166           <p>
              Aqui ponemos informacion de artefactosd e salida.
168           </p>
              <s:optiontransferselect
170                   id="artefactosProducidos"
                      name="artefactosProducidos"
172                       leftTitle="Artefactos Producidos"
                          rightTitle="Artefactos Disponibles"
174                       list="actividad.producidos"
                          listKey="nombre"
176                       listValue="nombre"
                          multiple="true"
178                       headerKey="0"
                          headerValue="---------- Artefactos Requeridos
                              ----------"
180                       size="6"
                          doubleId="artefactosDisponibles2"
182                       doubleList="artefactos"
                          doubleListKey="nombre"
184                       doubleListValue="nombre"
                          doubleName=""
186                       doubleHeaderKey="0"
                          doubleHeaderValue="<---------- Artefactos Disponibles
                              ---------->"
188                       doubleMultiple="true"
                          doubleSize="6"
190                       allowUpDownOnLeft="false"
                          allowUpDownOnRight="false"
192                       allowAddAllToLeft="false"
                          allowAddAllToRight="false"
194                       allowSelectAll="false"
                    />
196         </s:div>
198         <h3><a href="#">Informacion General</a></h3>
            <s:div>
200           <p>
              Aqui ponemos informacion realcionada.
202           </p>
              <tr>
204           <td colspan="2"><div id="descripcion" align="left">
              <textarea id="text1" class="wymeditor" name="actividad.
                  informacionGeneral">
206
              </textarea>
```

```
208         <input type="button" value="Aceptar" class="wymupdate" />
            </div>
210         </td>
            </tr>
212       </s:div>

214     </s:div>


216
        <td colspan="1"><div align="left"></div>
218       <input type="submit"
                name="enviar2"
220             value="Eliminar"
            onclick=this.form.action="">
222       </td>
          <td colspan="3"><div align="right">
224       <input type="submit"
                name="enviar2"
226             value="Guardar"
            onclick=this.form.action="guardarActividadModificada">
228         </div>
        </td>
230

    </form>
232   <s:a href="actividadesRup">Regresar</s:a>


234
<!-- ############################################################ -->
236   </div>
      <div id="right-column">
238     <strong class="h">INFO</strong>
        <div class="box"> </div>
240   </div>
</div>
```

jsp/actividad/modificarEliminarActividad.jsp

```
1
<%@ taglib prefix="s" uri="/struts-tags" %>
3<div id="middle">
    <div id="left-column">
5     <h3>Menu</h3>
      <ul class="nav">
7       <li class=""><a href="<s:url action="actividadesRup"/>">
            Actividades (RUP)</a></li>
        <li class=""><a href="#">Actividades del Proyecto</a></li>
9       <li class=""><a href="<s:url action="nuevaActividad"/>">Nuevo
            Actividad</a></li>
      </ul>
11    <a href="<s:url action="salirDelSistema"></s:url>" class="link">
          Salir </a>
    </div>
13
```

```
    <div id="center-column">
<!-- ######################################################### -->
    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Actividad Agregada</h1>
        <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
            Proyectos</a></div>
    </div><br/>

    Nombre: <s:property value="actividad.nombre"/><br></br>

    Disciplina: <s:property value="actividad.disciplina.nombre"/><br></
        br>

    Rol: <s:property value="actividad.rol.nombreRol"/><br></br>

    Descripcion: <s:property value="actividad.descripcion" escape="false
        "/><br></br>

    Descripcion: <s:property value="actividad.informacionGeneral" escape
        ="false"/><br></br>

    Artefactos Requeridos:<br></br>
    <s:iterator value="actividad.requeridos" var="it">
      <s:property value="nombre"/><br></br>
    </s:iterator><br></br>

    Artefactos Producidos:<br></br>

    <s:iterator value="actividad.producidos" var="it">
      <s:property value="nombre"/><br></br>
    </s:iterator><br></br>

    <s:a href="nuevaActividad">Regresar</s:a>


<!-- ######################################################### -->
    </div>
    <div id="right-column">
      <strong class="h">INFO</strong>
      <div class="box"></div>
    </div>
</div>
```

jsp/actividad/actividadAgregada.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>
<div id="middle">
    <div id="left-column">
      <h3>Menu</h3>
      <ul class="nav">
```

```
        <li class=""><a href="<s:url action="actividadesRup"/>">
            Actividades (RUP)</a></li>
        <li class=""><a href="#">Actividades del Proyecto</a></li>
        <li class=""><a href="<s:url action="nuevaActividad"/>">Nuevo
            Actividad</a></li>
      </ul>
      <a href="<s:url action="salirDelSistema"></s:url>" class="link">
         Salir</a>
    </div>

    <div id="center-column">
<!-- ############################################################ -->
    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Actividad: <s:property value="actividad.nombre"></s:property
            ></h1>
        <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
            Proyectos</a></div>
    </div><br/>

    Descripcion: <s:property value="actividad.descripcion" escape="false
        "/><br></br>

    Rol: <s:property value="actividad.rol.nombreRol"/>

    Descripcion: <s:property value="actividad.informacionGeneral" escape
        ="false"/><br></br>


    <s:a href="actividadesRup">Regresar</s:a>


<!-- ############################################################ -->
    </div>
    <div id="right-column">
      <strong class="h">INFO</strong>
      <div class="box"></div>
    </div>
</div>
```

jsp/actividad/actividadModificada.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>
<link type="text/css" href="/PrototipoWeb/jsp/css/jquery-ui-1.8.2.custom
    .css" rel="Stylesheet"/>
<script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery.
    wymeditor.min.js"></script>
<script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery-ui
    -1.8.2.custom.min.js"></script>
<script type="text/javascript">
```

```
/* Here we replace each element with class 'wymeditor'
 * (typically textareas) by a WYMeditor instance.
 *
 * We could use the 'html' option, to initialize the editor's content.
 * If this option isn't set, the content is retrieved from
 * the element being replaced.
 */

var x = $(document);
x.ready(inicializarEventos);

function inicializarEventos(){
 var x = $(".wymeditor");
   x.wymeditor({
       //classes panel
      /* classesItems: [
         {'name': 'date', 'title': 'PARA: Date', 'expr': 'p'},
         {'name': 'hidden-note', 'title': 'PARA: Hidden note',
          'expr': 'p[@class!="important"]'},
         {'name': 'important', 'title': 'PARA: Important',
          'expr': 'p[@class!="hidden-note"]'},
         {'name': 'border', 'title': 'IMG: Border', 'expr': 'img'},
         {'name': 'special', 'title': 'LIST: Special', 'expr': 'ul, ol
             '}
      ],*/

      //we customize the XHTML structure of WYMeditor by overwriting
      //the value of boxHtml. In this example, "CONTAINERS" and
      //"CLASSES" have been moved from "wym_area_right" to "
          wym_area_top":
      boxHtml:    "<div class='wym_box'>"
              + "<div class='wym_area_top'>"
              + WYMeditor.TOOLS
              + WYMeditor.CONTAINERS
              + WYMeditor.CLASSES
              + "</div>"
              + "<div class='wym_area_left'></div>"
              + "<div class='wym_area_right'>"
              + "</div>"
              + "<div class='wym_area_main'>"
              + WYMeditor.HTML
              + WYMeditor.IFRAME
              + WYMeditor.STATUS
              + "</div>"
              + "<div class='wym_area_bottom'>"
              + "</div>"
              + "</div>",

      //postInit is a function called when WYMeditor instance is ready
      //wym is the WYMeditor instance
      postInit: function(wym) {
```

```
58            //we make all sections in area_top render as dropdown menus:
              jQuery(wym._box)
60                //first we have to select them:
                  .find(".wym_area_top .wym_section")
62                //then we remove the existing class which make some of
                      them render as a panels:
                  .removeClass("wym_panel")
64                //then we add the class which will make them render as a
                      dropdown menu:
                  .addClass("wym_dropdown")
66                //finally we add some css to make the dropdown menus look
                      better:
                  .css("width", "160px")
68                .css("float", "left")
                  .css("margin-right", "5px")
70                .find("ul")
                  .css("width", "140px");
72
              //add a ">" character to the title of the new dropdown menus (
                  visual cue)
74            jQuery(this._box).find(".wym_tools, .wym_classes ")
                  .find(WYMeditor.H2)
76                .append("<span> &gt;</span>");
              }
78        });


80
    var x = $("#acordeonActividad");
82  x.accordion({ autoHeight: false});
  }
84</script>

86<div id="middle">
      <div id="left-column">
88      <h3>Menu</h3>
        <ul class="nav">
90        <li class=""><a href="<s:url action="actividadesRup"/>">
              Actividades (RUP)</a></li>
          <li class=""><a href="#">Actividades del Proyecto</a></li>
92        <li class=""><a href="<s:url action="nuevaActividad"/>">Nueva
              Actividad</a></li>
        </ul>
94      <a href="<s:url action="salirDelSistema"></s:url>" class="link">
          Salir</a>
      </div>
96
      <div id="center-column">
98<!-- ################################################################ -->
      <div class="top-bar">
100       <a href="<s:url action="salirDelSistema"></s:url>" class="button
              ">Salir </a>
          <h1>Nueva Actividad</h1>
```

```
102         <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
                Proyectos</a></div>
        </div><br></br>
104
        <p>Actividad</p>
106
        <form action="agregarActividad">
108
        <s:textfield label="Nombre" name="actividad.nombre"></s:textfield><
            br></br>
110     <div>
        <s:select name="nombreRol" list="roles" listKey="nombreRol"
112       listValue="nombreRol" label="Rol">
        </s:select><br></br>
114     </div>
        <s:select name="idDisciplina" list="disciplinas" listKey="
            idDisciplina"
116       listValue="nombre" label="Disciplina">
        </s:select>
118


120


122     <br></br>
        <br></br>
124     <s:div id="acordeonActividad">
          <h3><a href="#">Proposito</a></h3>
126       <s:div>
            <p>Edite la informacion y a continuacion presione Aceptar</p>
128         <tr>
            <td colspan="2"><div id="descripcion" align="left">
130         <textarea id="text1" class="wymeditor" name="actividad.
                descripcion">
              <s:property value="actividad.descripcion"/>
132         </textarea>
            <input type="button" value="Aceptar" class="wymupdate" />
134         </div>
            </td>
136         </tr>
          </s:div>
138
          <h3><a href="#">Artefactos Requeridos</a></h3>
140       <s:div>
            <p>
142         Artefactos que se necesitan para realizar esta actividad.
            </p>
144         <div>
                <s:optiontransferselect
146                 id="artefactosEntrada"
                    name="artefactosEntrada"
148                     leftTitle="Artefactos Requeridoss"
                        rightTitle="Artefactos Disponibles"
```

```
150                     list="{}"
                        listKey="nombre"
152                     listValue="nombre"
                        multiple="true"
154                     headerKey="0"
                        headerValue="---------- Artefactos Requeridos
                            ----------"
156                     size="6"
                        doubleId="artefactosDisponibles1"
158                     doubleList="artefactos"
                        doubleListKey="nombre"
160                     doubleListValue="nombre"
                        doubleName=""
162                     doubleHeaderKey="0"
                        doubleHeaderValue="<---------- Artefactos Disponibles
                            ---------->"
164                     doubleMultiple="true"
                        doubleSize="6"
166                     allowUpDownOnLeft="false"
                        allowUpDownOnRight="false"
168                     allowAddAllToLeft="false"
                        allowAddAllToRight="false"
170                     allowSelectAll="false"
                />
172       </div>
      </s:div>
174
      <h3><a href="#">Artefactos Producidos</a></h3>
176   <s:div>
        <p>
178     Artefactos que son generados durante esta actividad.
        </p>
180     <div>
            <s:optiontransferselect
182             id="artefactosProducidos"
                name="artefactosProducidos"
184                 leftTitle="Artefactos Producidos"
                    rightTitle="Artefactos Disponibles"
186                 list="{}"
                    listKey="nombre"
188                 listValue="nombre"
                    multiple="true"
190                 headerKey="0"
                    headerValue="---------- Artefactos Requeridos
                        ----------"
192                 size="6"
                    doubleId="artefactosDisponibles2"
194                 doubleList="artefactos"
                    doubleListKey="nombre"
196                 doubleListValue="nombre"
                    doubleName=""
198                 doubleHeaderKey="0"
```

```
                    doubleHeaderValue="<--------- Artefactos Disponibles
                        ------>"
                    doubleMultiple="true"
                    doubleSize="6"
                    allowUpDownOnLeft="false"
                    allowUpDownOnRight="false"
                    allowAddAllToLeft="false"
                    allowAddAllToRight="false"
                    allowSelectAll="false"
                />
        </div>
      </s:div>

      <h3><a href="#">Informacion General</a></h3>
      <s:div>
        <p>
        Escriba y despues presione Aceptar.
        </p>
        <tr>
        <td colspan="2"><div id="descripcion" align="left">
        <textarea id="text1" class="wymeditor" name="actividad.
            informacionGeneral">

        </textarea>
        <input type="button" value="Aceptar" class="wymupdate" />
        </div>
        </td>
        </tr>
      </s:div>

    </s:div>
    <s:submit value="Guardar"></s:submit>
  </form>
    <s:a href="actividadesRup">Regresar</s:a>


<!-- ############################################################ -->
    </div>
    <div id="right-column">
      <strong class="h">INFO</strong>
      <div class="box"> </div>
    </div>
</div>
```

jsp/actividad/nuevaActividad.jsp

```

<%@ taglib prefix="s" uri="/struts-tags" %>
<div id="middle">
    <div id="left-column">
      <h3>Menu</h3>
      <ul class="nav">
```

```
7           <li class=""><a href="<s:url action="actividadesRup"/>">
                Actividades (RUP)</a></li>
            <li class=""><a href="#">Actividades del Proyecto</a></li>
9           <li class=""><a href="<s:url action="nuevaActividad"/>">Nuevo
                Actividad</a></li>
        </ul>
11      <a href="<s:url action="salirDelSistema"></s:url>" class="link">
            Salir</a>
    </div>
13
    <div id="center-column">
15<!-- ############################################################ -->
    <div class="top-bar">
17      <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Actividad: <s:property value="actividad.nombre"></s:property
            ></h1>
19      <div class="breadcrumbs"><a href="#">Actividad</a> / <a href="#"
            >Detalles Actividad</a></div>
    </div><br/>
21
    <h3>Actividad</h3>
23  <s:property value="actividad.nombre"></s:property>

25  <h3>Descripcion</h3>
    <s:property value="actividad.descripcion"/>
27
    <h3>Rol</h3>
29  <s:property value="actividad.ron.nombreRol"/>

31  <h3>Disciplina</h3>
    <s:property value="actividad.disciplina.nombre"/>
33
    <h3>Informacion General</h3>
35  <s:property value="actividad.informacionGeneral"/>

37  <h3>Artefactos Requeridos</h3>
    <s:iterator value="actividad.requeridos" var="it">
39    <s:property value="nombre"/><br></br>
    </s:iterator>
41
    <h3>Artefactos Producidos</h3>
43  <s:iterator value="actividad.producidos" var="it">
      <s:property value="nombre"/><br></br>
45  </s:iterator>

47  <s:a href="actividadesRup">Regresar</s:a>

49
<!-- ############################################################ -->
51  </div>
    <div id="right-column">
```

```
      <strong class="h">INFO</strong>
      <div class="box"></div>
    </div>
</div>
```

jsp/actividad/detallesActividad.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>
  <div id="header">
    <a href="index.html" class="logo"><img src="img/logo.gif" width="101
        " height="29" alt="" /></a>
    <ul id="top-navigation">
      <li><span><span><a href="<s:url action="inicio"/>">Inicio</a></
          span></span></li>
      <li><span><span><a href="<s:url action="proyectosUsuario"/>">
          Proyecto</a></span></span></li>
      <li><span><span><a href="<s:url action="personalProyecto"/>">
          Personal</a></span></span></li>
      <li><span><span><a href="<s:url action="iteraciones"/>">Planificar
          </a></span></span></li>
      <li  class="active"><span><span><a href="actividadesRup">
          Actividades</a></span></span></li>
      <li><span><span><a href="rolesSistema">Rol</a></span></span></li>
      <li><span><span><a href="artefactosSistema">Artefactos</a></span
          ></span></li>
      <li><span><span><a href="tareasUsuario">Tareas</a></span></span></
          li>
      <li><span><span><a href="verPermisos">Seguridad</a></span></span
          ></li>
    </ul>
  </div>
```

jsp/actividad/header.jsp

```

<%@ taglib prefix="s" uri="/struts-tags" %>
<div id="middle">
    <div id="left-column">
      <h3>Menu</h3>
      <ul class="nav">
        <li class=""><a href="<s:url action="actividadesRup"/>">
            Actividades (RUP)</a></li>
        <li class=""><a href="#">Actividades del Proyecto</a></li>
        <li class=""><a href="<s:url action="nuevaActividad"/>">Nuevo
            Actividad</a></li>
      </ul>
      <a href="<s:url action="salirDelSistema"></s:url>" class="link">
          Salir</a>
    </div>

    <div id="center-column">
<!-- ######################################################### -->
    <div class="top-bar">
```

```
17          <a href="<s:url action="salirDelSistema"></s:url>" class="button
                ">Salir </a>
            <h1>Actividades definidas por RUP</h1>
19          <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
                Proyectos</a></div>
        </div><br/>
21
        <p>Actividades de RUP</p>
23
        <s:iterator value="actividades" var="it">
25        <!-- Agregamos una liga para que el usuario
            vea la informacion detallada del rol asociado al artefaco -->
27        <s:url id="dt1" action="detallesRol">
              <s:param name="nombreRol">
29              <s:property value="rol.nombreRol"/>
              </s:param>
31        </s:url>

33        <!-- Agregamos una liga para que el usuario
            vea la informacion detallada del artefacto -->
35        <s:url id="dt" action="detallesActividad">
              <s:param name="idActividad">
37              <s:property value="idActividad"/>
              </s:param>
39        </s:url>

41        <!-- Agregamos una liga para que el usuario
            pueda modificar el artefacto -->
43        <s:url id="dt2" action="modificarEliminarActividad">
              <s:param name="idActividad">
45              <s:property value="idActividad"/>
              </s:param>
47        </s:url>

49        <s:if test="disciplina.idDisciplina == 1">
            <s:a href="%{dt}"><s:property value="nombre"/></s:a><br></br>
51          <s:a href="%{dt1}"><s:property value="rol.nombreRol"/></s:a><br
                ></br>
            <s:a href="%{dt2}">Modificar/Eliminar</s:a><br></br><br></br>
53        </s:if><s:else>
            <s:a href="%{dt}"><s:property value="nombre"/></s:a><br></br>
55          <s:a href="%{dt1}"><s:property value="rol.nombreRol"/></s:a><br
                ></br>
            <s:a href="%{dt2}">Modificar/Eliminar</s:a><br></br><br></br>
57        </s:else>
          <br></br>
59


61      </s:iterator>

63  <!-- ################################################################ -->
        </div>
```

```
65      <div id="right-column">
          <strong class="h">INFO</strong>
67        <div class="box"> </div>
        </div>
69  </div>
```

jsp/actividad/actividadesRup.jsp

```
1
  <%@ taglib prefix="s" uri="/struts-tags" %>
3 <div id="middle">
      <div id="left-column">
5       <h3>Menu</h3>
        <ul class="nav">
7         <li class=""><a href="<s:url action="artefactosSistema"/>">
              Artefactos (RUP)</a></li>
          <li class=""><a href="#">Artefactos del Proyecto</a></li>
9         <li class=""><a href="<s:url action="nuevoArtefacto"/>">Nuevo
              Artefacto</a></li>
          <li class=""><a href="<s:url action="modificarEliminarArtefacto"
              />">Modificar/Eliminar Artefacto</a></li>
11      </ul>
        <a href="<s:url action="salirDelSistema"></s:url>" class="link">
            Salir</a>
13    </div>

15    <div id="center-column">
  <!-- ##################################################################### -->
17    <div class="top-bar">
          <a href="<s:url action="salirDelSistema"></s:url>" class="button
              ">Salir </a>
19        <h1>Informacion del Artefacto</h1>
          <div class="breadcrumbs"><a href="#">Artefacto</a> / <a href="#"
              >Detalles Artefacto</a></div>
21    </div><br/>

23    <h3>Artefacto</h3>
      <s:property value="artefacto.nombre"/><br></br>
25
      <h3>Rol</h3>
27    <s:property value="artefacto.rol.nombreRol"/><br></br>

29    <h3>Descripción</h3>
      <s:property value="artefacto.descripcion" escape="false"/><br></br>
31


33    <h3>Representación UML</h3>
      <s:property value="artefacto.representacionUML" escape="false"/><br
          ></br>
35
      <h3>Propósito</h3>
37    <s:property value="artefacto.proposito" escape="false"/><br></br>
```

```
39      <h3>Momento de Creación</h3>
        <s:property value="artefacto.timing" escape="false"/><br></br>
41
        <h3>Adaptación</h3>
43      <s:property value="artefacto.tailoring" escape="false"/><br></br>

45      <br></br>
        <br></br>
47      <a href="<s:url action="artefactosSistema"/>">Regresar</a>

49  <!-- ################################################################ -->
        </div>
51      <div id="right-column">
          <strong class="h">INFO</strong>
53        <div class="box"></div>
        </div>
55  </div>
```

jsp/artefactos/detallesArtefacto.jsp

```
1
  <%@ taglib prefix="s" uri="/struts-tags" %>
3 <div id="middle">
        <div id="left-column">
5         <h3>Menu</h3>
          <ul class="nav">
7           <li class=""><a href="<s:url action="artefactosSistema"/>">
                Artefactos (RUP)</a></li>
            <li class=""><a href="#">Artefactos del Proyecto</a></li>
9           <li class=""><a href="<s:url action="nuevoArtefacto"/>">Nuevo
                Artefacto</a></li>
            <li class=""><a href="<s:url action="modificarEliminarArtefacto"
                />">Modificar/Eliminar Artefacto</a></li>
11        </ul>
          <a href="<s:url action="salirDelSistema"></s:url>" class="link">
              Salir</a>
13      </div>

15      <div id="center-column">
  <!-- ################################################################ -->
17      <div class="top-bar">
            <a href="<s:url action="salirDelSistema"></s:url>" class="button
                ">Salir </a>
19          <h1>Informacion del Rol</h1>
            <div class="breadcrumbs"><a href="#">Artefactos</a> / <a href="#
                ">Detalles Rol</a></div>
21      </div><br/>

23      Nombre: <s:property value="rol.nombreRol"/><br></br>

25      Descripcion: <s:property value="rol.descripcion"/><br></br>

27      Categoria: <s:property value="rol.categoria"/><br></br>
```

```
29      Habilidades: <s:property value="rol.habilidades"/><br></br>

31      Criterio asignacion: <s:property value="rol.criteriosAsignacion"/><
            br></br>

33      <a href="<s:url action="artefactosSistema"/>">Regresar</a>

35  <!-- ############################################################ -->
        </div>
37      <div id="right-column">
          <strong class="h">INFO</strong>
39        <div class="box"></div>
        </div>
41  </div>
```

jsp/artefactos/detallesRol.jsp

```
1
  <%@ taglib prefix="s" uri="/struts-tags" %>
3 <link type="text/css" href="/PrototipoWeb/jsp/css/jquery-ui-1.8.2.custom
      .css" rel="Stylesheet"/>
  <script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery.
      wymeditor.min.js"></script>
5 <script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery-ui
      -1.8.2.custom.min.js"></script>
  <script type="text/javascript">
7
  /* Here we replace each element with class 'wymeditor'
9  * (typically textareas) by a WYMeditor instance.
   *
11 * We could use the 'html' option, to initialize the editor's content.
   * If this option isn't set, the content is retrieved from
13 * the element being replaced.
   */
15
   var x = $(document);
17 x.ready(inicializarEventos);

19 function inicializarEventos(){
    var x = $(".wymeditor");
21    x.wymeditor({
          //classes panel
23        /* classesItems: [
            {'name': 'date', 'title': 'PARA: Date', 'expr': 'p'},
25          {'name': 'hidden-note', 'title': 'PARA: Hidden note',
              'expr': 'p[@class!="important"]'},
27          {'name': 'important', 'title': 'PARA: Important',
              'expr': 'p[@class!="hidden-note"]'},
29          {'name': 'border', 'title': 'IMG: Border', 'expr': 'img'},
            {'name': 'special', 'title': 'LIST: Special', 'expr': 'ul, ol
                '}
31        ],*/
```

```
33          //we customize the XHTML structure of WYMeditor by overwriting
            //the value of boxHtml. In this example, "CONTAINERS" and
35          //"CLASSES" have been moved from "wym_area_right" to "
                wym_area_top":
            boxHtml:    "<div class='wym_box'>"
37                  + "<div class='wym_area_top'>"
                    + WYMeditor.TOOLS
39                  + WYMeditor.CONTAINERS
                    + WYMeditor.CLASSES
41                  + "</div>"
                    + "<div class='wym_area_left'></div>"
43                  + "<div class='wym_area_right'>"
                    + "</div>"
45                  + "<div class='wym_area_main'>"
                    + WYMeditor.HTML
47                  + WYMeditor.IFRAME
                    + WYMeditor.STATUS
49                  + "</div>"
                    + "<div class='wym_area_bottom'>"
51                  + "</div>"
                    + "</div>",
53
            //postInit is a function called when WYMeditor instance is ready
55          //wym is the WYMeditor instance
            postInit: function(wym) {
57
              //we make all sections in area_top render as dropdown menus:
59            jQuery(wym._box)
                  //first we have to select them:
61                .find(".wym_area_top .wym_section")
                  //then we remove the existing class which make some of
                      them render as a panels:
63                .removeClass("wym_panel")
                  //then we add the class which will make them render as a
                      dropdown menu:
65                .addClass("wym_dropdown")
                  //finally we add some css to make the dropdown menus look
                      better:
67                .css("width", "160px")
                  .css("float", "left")
69                .css("margin-right", "5px")
                  .find("ul")
71                .css("width", "140px");

73            //add a ">" character to the title of the new dropdown menus (
                  visual cue)
              jQuery(this._box).find(".wym_tools, .wym_classes ")
75                .find(WYMeditor.H2)
                  .append("<span> &gt;</span>");
77            }
        });
```

```
79
81    var x = $("#acordeonArtefacto");
      x.accordion({ autoHeight: false});
83   }
  </script>
85 <div id="middle">
       <div id="left-column">
87       <h3>Menu</h3>
         <ul class="nav">
89         <li class=""><a href="<s:url action="artefactosSistema"/>">
               Artefactos (RUP)</a></li>
           <li class=""><a href="#">Artefactos del Proyecto</a></li>
91         <li class=""><a href="<s:url action="nuevoArtefacto"/>">Nuevo
               Artefacto</a></li>
           <li class=""><a href="<s:url action="modificarEliminarArtefacto"
               />">Modificar/Eliminar Artefacto</a></li>
93       </ul>
         <a href="<s:url action="salirDelSistema"></s:url>" class="link">
             Salir</a>
95     </div>

97     <div id="center-column">
  <!-- ########################################################## -->
99     <div class="top-bar">
           <a href="<s:url action="salirDelSistema"></s:url>" class="button
               ">Salir </a>
101        <h1>Nuevo Artefacto</h1>
           <div class="breadcrumbs"><a href="#">Artefacto</a> / <a href="#"
               >Nuevo Artefacto</a></div>
103    </div><br/>

105    <p>Datos del nuevo artefacto</p>

107    <form action="guardarArtefacto" method="post">

109    <s:textfield label="Nombre" name="artefacto.nombre"></s:textfield><
           br></br><br></br>

111    <s:select name="rolElegido" list="roles" label="Rol"
         listKey="nombreRol" listValue="nombreRol">
113    </s:select><br></br><br></br>

115    <tr>
       <s:div id="acordeonArtefacto">
117      <h3><a href="#">Descripcion</a></h3>
         <div>
119        <textarea class="wymeditor" name="artefacto.descripcion">
           </textarea>
121        <input type="button" value="Aceptar" class="wymupdate" />
         </div>
123
```

```
      <h3><a  href="#">Representacion UML</a></h3>
125   <div>
        <textarea class="wymeditor" name="artefacto.representacionUML">
127     </textarea>
        <input type="button" value="Aceptar" class="wymupdate" />
129   </div>
      <h3><a href="#">Proposito</a></h3>
131   <div>
        <textarea class="wymeditor" name="artefacto.proposito">
133     </textarea>
        <input type="button" value="Aceptar" class="wymupdate" />
135   </div>
      <h3><a href="#">Momento de Creacion</a></h3>
137   <div>
        <textarea class="wymeditor" name="artefacto.timing">
139     </textarea>
        <input type="button" value="Aceptar" class="wymupdate" />
141   </div>
      <h3><a href="#">Adaptaciones</a></h3>
143   <div>
        <textarea class="wymeditor" name="artefacto.tailoring">
145     </textarea>
        <input type="button" value="Aceptar" class="wymupdate" />
147   </div>
    </s:div>
149 </tr>

151 <s:submit value="Guardar"></s:submit>
    </form>
153
<!-- ########################################################## -->
155 </div>
    <div id="right-column">
157   <strong class="h">INFO</strong>
      <div class="box"> </div>
159 </div>
</div>
```

jsp/artefactos/nuevoArtefacto.jsp

```
2<%@ taglib prefix="s" uri="/struts-tags" %>
<div id="middle">
4   <div id="left-column">
      <h3>Menu</h3>
6     <ul class="nav">
        <li class=""><a href="<s:url action="artefactosSistema"/>">
            Artefactos (RUP)</a></li>
8       <li class=""><a href="#">Artefactos del Proyecto</a></li>
        <li class=""><a href="<s:url action="nuevoArtefacto"/>">Nuevo
            Artefacto</a></li>
10      <li class=""><a href="<s:url action="modificarEliminarArtefacto"
            />">Modificar/Eliminar Artefacto</a></li>
```

```
      </ul>
      <a href="<s:url action="salirDelSistema"></s:url>" class="link">
          Salir</a>
    </div>

    <div id="center-column">
<!-- ############################################################ -->
    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Artefacto Eliminado</h1>
        <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
            Proyectos</a></div>
    </div><br/>

    Artefacto: <s:property value="artefacto.nombre"/><br></br>

    Descripcion: <s:property value="artefacto.descripcion"/><br></br>

    Rol: <s:property value="artefacto.rol.nombreRol"/><br></br>

    Representacion UML: <s:property value="artefacto.representacionUML"
       /><br></br>

    Proposito: <s:property value="artefacto.proposito"/><br></br>

    Timing: <s:property value="artefacto.proposito"/><br></br>

    Tailoring: <s:property value="artefacto.tailoring"/><br></br>

    <br></br>
    <br></br>
    <a href="<s:url action="artefactosSistema"/>">Regresar</a>

<!-- ############################################################ -->
    </div>
    <div id="right-column">
      <strong class="h">INFO</strong>
      <div class="box"></div>
    </div>
</div>
```

jsp/artefactos/artefactoEliminado.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>

<div id="middle">
    <div id="left-column">
      <h3>Menu</h3>
      <ul class="nav">
      </ul>
    </div>

```

```
    <div id="center-column">
<!-- ######################################################## -->
    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Plantilla Artefacto</h1>
        <div class="breadcrumbs"><a href="#">Plantilla</a> / <a href="#"
            >Plantillas</a>
        </div><br/>
    </div>
    <div class="contenidoPrincipal">
    </div>
<!-- ######################################################## -->
    </div>
    <div id="right-column">
      <strong class="h">INFO</strong>
      <div class="box"></div>
    </div>
</div>
```

jsp/artefactos/subirPlantilla.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>
<div id="middle">
    <div id="left-column">
      <h3>Menu</h3>
      <ul class="nav">
        <li class=""><a href="<s:url action="artefactosSistema"/>">
            Artefactos (RUP)</a></li>
        <li class=""><a href="#">Artefactos del Proyecto</a></li>
        <li class=""><a href="<s:url action="nuevoArtefacto"/>">Nuevo
            Artefacto</a></li>
        <li class=""><a href="<s:url action="modificarEliminarArtefacto"
            />">Modificar/Eliminar Artefacto</a></li>
      </ul>
      <a href="<s:url action="salirDelSistema"></s:url>" class="link">
          Salir</a>
    </div>

    <div id="center-column">
<!-- ######################################################## -->
    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Artefactos</h1>
        <div class="breadcrumbs"><a href="#">Artefactos</a> / <a href="#
            ">Artefactos (RUP)</a></div>
    </div><br/>

    <p>Artefactos de RUP</p>

    <s:iterator value="artefactos" var="it">
```

```
26        <!-- Agregamos una liga para que el usuario
            vea la informacion detallada del rol asociado al artefaco -->
28        <s:url id="dt1" action="detallesRol">
              <s:param name="nombreRol">
30              <s:property value="rol.nombreRol"/>
              </s:param>
32        </s:url>

34        <!-- Agregamos una liga para que el usuario
            vea la informacion detallada del artefacto -->
36        <s:url id="dt" action="detallesArtefacto">
              <s:param name="nombreArtefacto">
38              <s:property value="nombre"/>
              </s:param>
40        </s:url>

42        <!-- Agregamos una liga para que el usuario
            pueda modificar el artefacto -->
44        <s:url id="dt2" action="modificarEliminarArtefacto">
              <s:param name="nombreArtefacto">
46              <s:property value="nombre"/>
              </s:param>
48        </s:url>

50        <s:if test="rol.categoria == 'Programador'">
            <s:a href="%{dt}"><s:property value="nombre"/></s:a><br></br>
52          <s:a href="%{dt1}"><s:property value="rol.nombreRol"/></s:a><br
                ></br>
            <s:a href="%{dt2}">Modificar/Eliminar</s:a><br></br><br></br>
54        </s:if><s:else>
            <s:a href="%{dt}"><s:property value="nombre"/></s:a><br></br>
56          <s:a href="%{dt1}"><s:property value="rol.nombreRol"/></s:a><br
                ></br>
            <s:a href="%{dt2}">Modificar/Eliminar</s:a><br></br><br></br>
58        </s:else>
          <br></br>
60


62      </s:iterator>

64  <!-- ############################################################ -->
      </div>
66    <div id="right-column">
        <strong class="h">INFO</strong>
68      <div class="box"> </div>
      </div>
70  </div>
```

jsp/artefactos/artefactosSistema.jsp

```
2 <%@ taglib prefix="s" uri="/struts-tags" %>
  <div id="middle">
```

```
4       <div id="left-column">
          <h3>Menu</h3>
6         <ul class="nav">
            <li class=""><a href="<s:url action="artefactosSistema"/>">
                Artefactos (RUP)</a></li>
8           <li class=""><a href="#">Artefactos del Proyecto</a></li>
            <li class=""><a href="<s:url action="nuevoArtefacto"/>">Nuevo
                Artefacto</a></li>
10          <li class=""><a href="<s:url action="modificarEliminarArtefacto"
                />">Modificar/Eliminar Artefacto</a></li>
          </ul>
12        <a href="<s:url action="salirDelSistema"></s:url>" class="link">
              Salir</a>
        </div>
14
        <div id="center-column">
16  <!-- ############################################################ -->
        <div class="top-bar">
18          <a href="<s:url action="salirDelSistema"></s:url>" class="button
                ">Salir </a>
            <h1>Artefacto Agregado</h1>
20          <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
                Proyectos</a></div>
        </div><br/>
22
        Artefacto: <s:property value="artefacto.nombre"/>
24
        Rol: <s:property value="artefacto.rol.nombreRol"/>
26
        Descripcion: <s:property value="artefacto.descripcion" escape="false
            "/>
28
        Representacion UML: <s:property value="artefacto.representacionUML"
            escape="false"/>
30
        Proposito: <s:property value="artefacto.proposito" escape="false"/>
32
        Timing: <s:property value="artefacto.proposito" escape="false"/>
34
        Tailoring: <s:property value="artefacto.tailoring" escape="false"/>
36
        <br></br>
38      <br></br>
        <a href="<s:url action="nuevoArtefacto"/>">Regresar</a>
40  <!-- ############################################################ -->
        </div>
42      <div id="right-column">
          <strong class="h">INFO</strong>
44        <div class="box"></div>
        </div>
46  </div>
```

jsp/artefactos/artefactoAgregado.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>
  <div id="header">
    <a href="index.html" class="logo"><img src="img/logo.gif" width="101
        " height="29" alt="" /></a>
    <ul id="top-navigation">
      <li><span><span><a href="<s:url action="inicio"/>">Inicio</a></
          span></span></li>
      <li><span><span><a href="<s:url action="proyectosUsuario"/>">
          Proyecto</a></span></span></li>
      <li><span><span><a href="<s:url action="personalProyecto"/>">
          Personal</a></span></span></li>
      <li><span><span><a href="<s:url action="iteraciones"/>">Planificar
          </a></span></span></li>
      <li><span><span><a href="actividadesRup">Actividades</a></span></
          span></li>
      <li><span><span><a href="rolesSistema">Rol</a></span></span></li>
      <li class="active"><span><span><a href="artefactosSistema">
          Artefactos</a></span></span></li>
      <li><span><span><a href="tareasUsuario">Tareas</a></span></span></
          li>
      <li><span><span><a href="verPermisos">Seguridad</a></span></span
          ></li>
    </ul>
  </div>
```

jsp/artefactos/header.jsp

```

<%@ taglib prefix="s" uri="/struts-tags" %>
<div id="middle">
    <div id="left-column">
      <h3>Menu</h3>
      <ul class="nav">
        <li class=""><a href="<s:url action="artefactosSistema"/>">
            Artefactos (RUP)</a></li>
        <li class=""><a href="#">Artefactos del Proyecto</a></li>
        <li class=""><a href="<s:url action="nuevoArtefacto"/>">Nuevo
            Artefacto</a></li>
        <li class=""><a href="<s:url action="modificarEliminarArtefacto"
            />">Modificar/Eliminar Artefacto</a></li>
      </ul>
      <a href="<s:url action="salirDelSistema"></s:url>" class="link">
          Salir</a>
    </div>

    <div id="center-column">
<!-- ##################################################################### -->
    <div class="top-bar">
```

```
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Artefacto Modificado</h1>
        <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
            Proyectos</a></div>
    </div><br/>

    Artefacto: <s:property value="artefacto.nombre"/><br></br>

    Descripcion: <s:property value="artefacto.descripcion"/><br></br>

    Rol: <s:property value="artefacto.rol.nombreRol"/><br></br>

    Representacion UML: <s:property value="artefacto.representacionUML"
        /><br></br>

    Proposito: <s:property value="artefacto.proposito"/><br></br>

    Timing: <s:property value="artefacto.proposito"/><br></br>

    Tailoring: <s:property value="artefacto.tailoring"/><br></br>

    <br></br>
    <br></br>
    <a href="<s:url action="artefactosSistema"/>">Regresar</a>

<!-- ############################################################ -->
    </div>
    <div id="right-column">
      <strong class="h">INFO</strong>
      <div class="box"></div>
    </div>
</div>
```

jsp/artefactos/artefactoModificado.jsp

```

<%@ taglib prefix="s" uri="/struts-tags" %>
<link type="text/css" href="/PrototipoWeb/jsp/css/jquery-ui-1.8.2.custom
    .css" rel="Stylesheet"/>
<script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery.
    wymeditor.min.js"></script>
<script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery-ui
    -1.8.2.custom.min.js"></script>
<script type="text/javascript">

/* Here we replace each element with class 'wymeditor'
 * (typically textareas) by a WYMeditor instance.
 *
 * We could use the 'html' option, to initialize the editor's content.
 * If this option isn't set, the content is retrieved from
 * the element being replaced.
 */
```

```
15
var x = $(document);
17 x.ready(inicializarEventos);

19 function inicializarEventos(){
 var x = $(".wymeditor");
21   x.wymeditor({
       //classes panel
23     /* classesItems: [
         {'name': 'date', 'title': 'PARA: Date', 'expr': 'p'},
25       {'name': 'hidden-note', 'title': 'PARA: Hidden note',
          'expr': 'p[@class!="important"]'},
27       {'name': 'important', 'title': 'PARA: Important',
          'expr': 'p[@class!="hidden-note"]'},
29       {'name': 'border', 'title': 'IMG: Border', 'expr': 'img'},
         {'name': 'special', 'title': 'LIST: Special', 'expr': 'ul, ol
            '}
31     ],*/

33     //we customize the XHTML structure of WYMeditor by overwriting
       //the value of boxHtml. In this example, "CONTAINERS" and
35     //"CLASSES" have been moved from "wym_area_right" to "
          wym_area_top":
       boxHtml:   "<div class='wym_box'>"
37             + "<div class='wym_area_top'>"
               + WYMeditor.TOOLS
39             + WYMeditor.CONTAINERS
               + WYMeditor.CLASSES
41             + "</div>"
               + "<div class='wym_area_left'></div>"
43             + "<div class='wym_area_right'>"
               + "</div>"
45             + "<div class='wym_area_main'>"
               + WYMeditor.HTML
47             + WYMeditor.IFRAME
               + WYMeditor.STATUS
49             + "</div>"
               + "<div class='wym_area_bottom'>"
51             + "</div>"
               + "</div>",
53
       //postInit is a function called when WYMeditor instance is ready
55     //wym is the WYMeditor instance
       postInit: function(wym) {
57
         //we make all sections in area_top render as dropdown menus:
59       jQuery(wym._box)
           //first we have to select them:
61         .find(".wym_area_top .wym_section")
           //then we remove the existing class which make some of
              them render as a panels:
63         .removeClass("wym_panel")
```

```
                //then we add the class which will make them render as a
                    dropdown menu:
65              .addClass("wym_dropdown")
                //finally we add some css to make the dropdown menus look
                    better:
67              .css("width", "160px")
                .css("float", "left")
69              .css("margin-right", "5px")
                .find("ul")
71              .css("width", "140px");

73          //add a ">" character to the title of the new dropdown menus (
                visual cue)
            jQuery(this._box).find(".wym_tools, .wym_classes ")
75              .find(WYMeditor.H2)
                .append("<span> &gt;</span>");
77          }
        });
79

81    var x = $("#acordeonArtefacto");
      x.accordion({ autoHeight: false});
83  }
</script>
85
<div id="middle">
87      <div id="left-column">
          <h3>Menu</h3>
89        <ul class="nav">
            <li class=""><a href="<s:url action="artefactosSistema"/>">
                Artefactos (RUP)</a></li>
91          <li class=""><a href="#">Artefactos del Proyecto</a></li>
            <li class=""><a href="<s:url action="nuevoArtefacto"/>">Nuevo
                Artefacto</a></li>
93          <li class=""><a href="<s:url action="modificarEliminarArtefacto"
                />">Modificar/Eliminar Artefacto</a></li>
          </ul>
95        <a href="<s:url action="salirDelSistema"></s:url>" class="link">
              Salir</a>
      </div>
97
      <div id="center-column">
99  <!-- ############################################################ -->
      <div class="top-bar">
101       <a href="<s:url action="salirDelSistema"></s:url>" class="button
              ">Salir </a>
          <h1>Modificar/Eliminar Artefacto</h1>
103       <div class="breadcrumbs"><a href="#">Artefacto</a> / <a href="#"
              >Modificar/Eliminar Artefacto</a></div>
      </div><br/>
105
      <p>Datos del nuevo artefacto</p>
```

```
107
    <form>
109
    <s:textfield label="Nombre" name="artefacto.nombre"></s:textfield>
111
    <tr>
113 <td><p>Rol</p></td>
    <td colspan="2"><div align="left">
115 <select name="nombreRol">
      <s:iterator value="roles" var="it">
117     <s:if test="nombreRol==artefacto.rol.nombreRol">
        <option value='<s:property value="nombreRol"></s:property>'
            selected><s:property value="nombreRol"></s:property></
            option>
119     </s:if><s:else>
        <option value='<s:property value="nombreRol"></s:property>'><s
            :property value="nombreRol"></s:property></option>
121     </s:else>
      </s:iterator>
123 </select>
    </div>
125 </td>
    </tr>
127 <br></br>
    <br></br>
129 <s:div id="acordeonArtefacto">
      <h3><a href="#">Descripcion</a></h3>
131   <div>
        <textarea class="wymeditor" name="artefacto.descripcion">
133       <s:property value="artefacto.descripcion"/>
        </textarea>
135     <input type="button" value="Aceptar" class="wymupdate" />
      </div>
137
      <h3><a  href="#">Representacion UML</a></h3>
139   <div>
        <textarea class="wymeditor" name="artefacto.representacionUML">
141       <s:property value="artefacto.representacionUML"/>
        </textarea>
143     <input type="button" value="Aceptar" class="wymupdate" />
      </div>
145   <h3><a href="#">Proposito</a></h3>
      <div>
147     <textarea class="wymeditor" name="artefacto.proposito">
          <s:property value="artefacto.proposito"/>
149     </textarea>
        <input type="button" value="Aceptar" class="wymupdate" />
151   </div>
      <h3><a href="#">Momento de Creacion</a></h3>
153   <div>
        <textarea class="wymeditor" name="artefacto.timing">
155       <s:property value="artefacto.timing"/>
```

```
        </textarea>
157       <input type="button" value="Aceptar" class="wymupdate" />
        </div>
159     <h3><a href="#">Adaptaciones</a></h3>
        <div>
161       <textarea class="wymeditor" name="artefacto.tailoring">
            <s:property value="artefacto.tailoring"/>
163       </textarea>
          <input type="button" value="Aceptar" class="wymupdate" />
165     </div>
    </s:div>
167
      <td colspan="1"><div align="left"></div>
169     <input type="submit"
              name="enviar2"
171           value="Eliminar"
          onclick=this.form.action="eliminarArtefacto">
173     </td>
        <td colspan="3"><div align="right">
175     <input type="submit"
              name="enviar2"
177           value="Guardar"
          onclick=this.form.action="guardarCambiosArtefacto">
179       </div>
        </td>
181   </form>

183   <a href="<s:url action="artefactosSistema"/>">Regresar</a>

185 <!-- ############################################################ -->
    </div>
187 <div id="right-column">
      <strong class="h">INFO</strong>
189   <div class="box"> </div>
    </div>
191 </div>
```

jsp/artefactos/modificarEliminarArtefacto.jsp

```
1
<%@ taglib prefix="s" uri="/struts-tags" %>
3 <link type="text/css" href="/PrototipoWeb/jsp/css/jquery-ui-1.8.2.custom
    .css" rel="Stylesheet" />
  <script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery-1.4.2.
    min.js"></script>
5 <script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery-ui
    -1.8.2.custom.min.js"></script>
  <script type="text/javascript" src="/PrototipoWeb/jsp/js/md5-min.js"></
    script>
7
  <script type="text/javascript">
9 <!--
  var x= $(document);
```

```
x.ready(inicializarEventos);

function inicializarEventos(){
  var x=$("#pass1");
  x.change(cifrar);

  var y = $("#pass2");
  y.change(cifrar1);

  var z = $("#password1");
  z.attr("value",null);

  var a = $("#pass_word");
  a.attr("value",null);
}

function cifrar(){
  var x=$("#pass1");
  var p = hex_md5(x.attr("value"));
  x = $("#password1");
  x.attr("value",p);
  //lert(x.attr("value"));
}

function cifrar1(){
  var x=$("#pass2");
  var p = hex_md5(x.attr("value"));
  x = $("#pass_word");
  x.attr("value",p);
  //alert(x.attr("value"));
}
//-->
</script>


<div id="middle">
    <div id="left-column">
      <h3>Header</h3>
      <ul class="nav">
        <li class=""><a href="<s:url action="personalProyecto"/>">
            Personal del Proyecto</a></li>
        <li class=""><a href="<s:url action="participantesProyecto"/>">
            Agregar/Eliminar Personal</a></li>
        <li class=""><a href="<s:url action="modificarEliminarUsuario"/>
            ">Eliminar/Modificar</a></li>
        <li class=""><a href="<s:url action="nuevoUsuario"/>">Nuevo
            Personal</a></li>
      </ul>
    </div>

    <div id="center-column">
<!-- ##################################################### -->
```

```
59      <div class="top-bar">
            <a href="<s:url action="salirDelSistema"></s:url>" class="button
                ">Salir </a>
61          <h1>Nuevo Personal</h1>
            <div class="breadcrumbs"><a href="#">Personal</a> / <a href="#">
                Nuevo Personal</a></div>
63      </div><br/>

65      <p>Agregar un Nuevo Usuario</p>

67      <s:form action="registrarUsuario">
          <s:textfield name="usuario.idUsuario" label="*CURP"></s:textfield>
69        <s:textfield name="usuario.nombre" label="*Nombre"></s:textfield>
          <s:textfield name="usuario.ape_paterno" label="Apellido Paterno"
              ></s:textfield>
71        <s:textfield name="usuario.ape_materno" label="Apellido Materno"
              ></s:textfield>
          <s:textfield name="usuario.correo" label="*Correo"></s:textfield>
73        <s:textfield name="usuario.telefono" label="Telefono"></s:
              textfield>
          <s:textfield name="usuario.login" label="Login"></s:textfield>
75
          <s:password  id="pass1" label="*Password"></s:password>
77        <s:hidden id="password1" name="password1"></s:hidden>

79        <s:password  id="pass2" label="*Repita el Password"></s:password>
          <s:hidden id="pass_word" name="usuario.pass_word"></s:hidden>
81



83


85        <s:optiontransferselect label="Rol del Nuevo Usuario"
            name="rolesAsignados"
87              leftTitle="Rol Asignado"
                rightTitle="Rol Disponible"
89              list="{'NINGUNO'}"
                listKey="nombreRol"
91              listValue="nombreRol"
                multiple="true"
93              headerKey="headerKey"
                headerValue="--- Rol Asignado ---"
95              size="5"
                emptyOption="false"
97              doubleList="roles"
                doubleListKey="nombreRol"
99              doubleListValue="nombreRol"
                doubleName=""
101             doubleHeaderKey="doubleHeaderKey"
                doubleMultiple="true"
103             doubleSize="5"
                allowUpDownOnLeft="false"
105             allowUpDownOnRight="false"
```

```
            allowAddAllToLeft="false"
            allowAddAllToRight="false"
            allowSelectAll="false"
    />


      <s:reset value="Reset"></s:reset><s:submit value="Registrar"></s:
          submit>
    </s:form>

    <a href="<s:url action="personalProyecto"/>">Regresar</a>
<!-- ############################################################ -->
    </div>
    <div id="right-column">
      <strong class="h">INFO</strong>
      <div class="box"> </div>
    </div>
</div>
```

jsp/personal/nuevoUsuario.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>
<div id="middle">
    <div id="left-column">
      <h3>Header</h3>
      <ul class="nav">
        <li class=""><a href="<s:url action="personalProyecto"/>">
            Personal del Proyecto</a></li>
        <li class=""><a href="<s:url action="participantesProyecto"/>">
            Agregar/Eliminar Personal</a></li>
        <li class=""><a href="<s:url action="modificarEliminarUsuario"/>
            ">Eliminar/Modificar</a></li>
        <li class=""><a href="<s:url action="nuevoUsuario"/>">Nuevo
            Personal</a></li>
      </ul>
    </div>

    <div id="center-column">
<!-- ############################################################ -->
    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Personal Eliminado</h1>
        <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
            Proyectos</a></div>
    </div><br/>

    <p>
    <s:actionmessage/>
    </p>

    <p>Datos del usuario: </p>
```

```
27
    <s:property value="usuario.idUsuario"/>
29  <s:property value="usuario.nombre"/>
    <s:property value="usuario.ape_paterno"/>
31  <s:property value="usuario.ape_materno"/>
    <s:property value="usuario.correo"/>
33  <s:property value="usuario.telefono"/>
    <s:property value="usuario.login"/>
35  <s:property value="usuario.pass_word"/>

37  <a href="<s:url action="modificarEliminarUsuario"/>">Regresar</a>
<!-- ############################################################ -->
39  </div>
    <div id="right-column">
41    <strong class="h">INFO</strong>
      <div class="box"> </div>
43  </div>
</div>
```

jsp/personal/usuarioEliminado.jsp

```
1
<%@ taglib prefix="s" uri="/struts-tags" %>
3<div id="middle">
    <div id="left-column">
5     <h3>Menu</h3>
      <ul class="nav">
7       <li class=""><a href="<s:url action="personalProyecto"/>">
            Personal del Proyecto</a></li>
        <li class=""><a href="<s:url action="participantesProyecto"/>">
            Agregar/Eliminar Personal</a></li>
9       <li class=""><a href="<s:url action="modificarEliminarUsuario"/>
            ">Eliminar/Modificar</a></li>
        <li class=""><a href="<s:url action="nuevoUsuario"/>">Nuevo
            Personal</a></li>
11    </ul>
    </div>
13
    <div id="center-column">
15<!-- ############################################################ -->
    <div class="top-bar">
17      <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Personal Registrado</h1>
19      <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
            Proyectos</a></div>
    </div><br/>
21
    El usuario se ha registrado con los siguientes datos:
23  <p>
    CURP: <s:property value="usuario.idUsuario"/>
25  </p>
    <p>
```

```
27      Nombre: <s:property value="usuario.nombre"/>
        </p>
29      <p>
        Apellido Paterno: <s:property value="usuario.ape_paterno"/>
31      </p>
        <p>
33      Apellido Materno: <s:property value="usuario.ape_materno"/>
        </p>
35      <p>
        Correo: <s:property value="usuario.correo"/>
37      </p>
        <p>
39      Telefono: <s:property value="usuario.telefono"/>
        </p>
41      <p>
        Login: <s:property value="usuario.login"/>
43      </p>
        <p>
45      Password: <s:property value="usuario.pass_word"/>
        </p>
47
        <p>Rol</p>
49      <s:iterator value="usuario.roles" var="it">
          <s:property value="nombreRol"/>
51      </s:iterator>

53      <a href="<s:url action="nuevoUsuario"/>">Regresar</a>
<!-- ############################################################ -->
55      </div>
        <div id="right-column">
57        <strong class="h">INFO</strong>
          <div class="box"></div>
59      </div>
</div>
```

jsp/personal/usuarioRegistrado.jsp

```
1
<%@ taglib prefix="s" uri="/struts-tags" %>
3<div id="middle">
        <div id="left-column">
5         <h3>Header</h3>
          <ul class="nav">
7           <li class=""><a href="<s:url action="personalProyecto"/>">
                Personal del Proyecto</a></li>
            <li class=""><a href="<s:url action="participantesProyecto"/>">
                Agregar/Eliminar Personal</a></li>
9           <li class=""><a href="<s:url action="modificarEliminarUsuario"/>
                ">Eliminar/Modificar</a></li>
            <li class=""><a href="<s:url action="nuevoUsuario"/>">Nuevo
                Personal</a></li>
11        </ul>
```

```
      <a href="<s:url action="salirDelSistema"></s:url>" class="link">
          Salir</a>
13    </div>

15    <div id="center-column">
<!-- ############################################################ -->
17    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
19      <h1>Agregar Participantes al Proyecto</h1>
        <div class="breadcrumbs"><a href="#">Personal</a> / <a href="#">
            Agregar/Eliminar Personal</a></div>
21    </div><br/>
    <p>Personal del Proyecto</p>
23    <s:form action="asignarPersonalProyecto">
      <s:optiontransferselect
25      name="usuariosElegidos"
            leftTitle="Personal Del Proyecto"
27          rightTitle="Personal Disponible"
            list="proyecto.listaUsuarios"
29          listKey="idUsuario"
            listValue="nombreCompleto"
31          multiple="true"
            headerKey="headerKey"
33          headerValue="--- Seleccione el Personal ---"
            size="12"
35          emptyOption="false"
            doubleList="usuariosSistema"
37          doubleListKey="idUsuario"
            doubleListValue="nombreCompleto"
39          doubleName=""
            doubleHeaderKey="doubleHeaderKey"
41          doubleMultiple="true"
            doubleSize="12"
43          allowUpDownOnLeft="false"
            allowUpDownOnRight="false"
45          allowAddAllToLeft="false"
            allowAddAllToRight="false"
47          allowSelectAll="false"
    />
49  <s:submit value="Guardar Cambios"></s:submit>

51  </s:form>

53  <a href="<s:url action="personalProyecto"/>">Regresar</a>
<!-- ############################################################ -->
55  </div>
    <div id="right-column">
57    <strong class="h">INFO</strong>
      <div class="box"></div>
59  </div>
</div>
```

jsp/personal/asignarParticipantesProyecto.jsp

```
1
<%@ taglib prefix="s" uri="/struts-tags" %>
3<link type="text/css" href="/PrototipoWeb/jsp/css/jquery-ui-1.8.2.custom
    .css" rel="Stylesheet" />
<script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery-1.4.2.
    min.js"></script>
5<script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery-ui
    -1.8.2.custom.min.js"></script>
<script type="text/javascript" src="/PrototipoWeb/jsp/js/md5-min.js"></
    script>
7
<script type="text/javascript">
9<!--
var x= $(document);
11x.ready(inicializarEventos);

13function inicializarEventos(){
   var x=$("#pass1");
15   x.change(cifrar);

17   var y = $("#pass2");
   y.change(cifrar1);
19
   var z = $("#password1");
21   z.attr("value",null);

23   var a = $("#pass_word");
   a.attr("value",null);
25}

27function cifrar(){
   var x=$("#pass1");
29   var p = hex_md5(x.attr("value"));
   x = $("#password1");
31   x.attr("value",p);
   //lert(x.attr("value"));
33}

35function cifrar1(){
   var x=$("#pass2");
37   var p = hex_md5(x.attr("value"));
   x = $("#pass_word");
39   x.attr("value",p);
   //alert(x.attr("value"));
41}
//-->
43</script>

45
```

```
<div id="middle">
47    <div id="left-column">
        <h3>Header</h3>
49      <ul class="nav">
          <li class=""><a href="<s:url action="personalProyecto"/>">
              Personal del Proyecto</a></li>
51        <li class=""><a href="<s:url action="participantesProyecto"/>">
              Agregar/Eliminar Personal</a></li>
          <li class=""><a href="<s:url action="modificarEliminarUsuario"/>
              ">Eliminar/Modificar</a></li>
53        <li class=""><a href="<s:url action="nuevoUsuario"/>">Nuevo
              Personal</a></li>
        </ul>
55    </div>

57    <div id="center-column">
<!-- ############################################################ -->
59    <div class="top-bar">
          <a href="<s:url action="salirDelSistema"></s:url>" class="button
              ">Salir </a>
61        <h1>Modificar/Eliminar Personal</h1>
          <div class="breadcrumbs"><a href="#">Personal</a> / <a href="#">
              Modificar/Eliminar Personal</a></div>
63    </div><br/>
      <br></br>
65    <p>Seleccion un usuario</p>
      <s:form id="form1" action="datosUsuario">
67      <s:select label="Usuario" name="idUsuario" list="proyecto.
            listaUsuarios" listKey="idUsuario"
          listValue="nombreCompleto"></s:select>
69      <s:submit value="Consultar Datos"></s:submit>
      </s:form>
71
      <p>Datos del Usuario</p>
73
      <s:fielderror></s:fielderror>
75    <s:form id="form2">

77      <s:hidden name="usuario.idUsuario" label="*CURP"></s:hidden>
        <s:textfield name="usuario.nombre" label="*Nombre"></s:textfield>
79      <s:textfield name="usuario.ape_paterno" label="Apellido Paterno"
            ></s:textfield>
        <s:textfield name="usuario.ape_materno" label="Apellido Materno"
            ></s:textfield>
81      <s:textfield name="usuario.correo" label="*Correo"></s:textfield>
        <s:textfield name="usuario.telefono" label="Telefono"></s:
            textfield>
83      <s:textfield name="usuario.login" label="Login"></s:textfield>

85      <s:optiontransferselect label="Rol del Nuevo Usuario"
          name="rolesAsignados"
87            leftTitle="Rol Asignado"
```

```
                rightTitle="Rol Disponible"
89              list="usuario.roles"
                listKey="nombreRol"
91              listValue="nombreRol"
                multiple="true"
93              headerKey="headerKey"
                headerValue="--- Rol Asignado ---"
95              size="5"
                emptyOption="false"
97              doubleList="rolesSistema"
                doubleListKey="nombreRol"
99              doubleListValue="nombreRol"
                doubleName=""
101             doubleHeaderKey="doubleHeaderKey"
                doubleMultiple="true"
103             doubleSize="5"
                allowUpDownOnLeft="false"
105             allowUpDownOnRight="false"
                allowAddAllToLeft="false"
107             allowAddAllToRight="false"
                allowSelectAll="false"
109         />
          <td colspan="3"><div align="right">
111         <input type="submit"
                 name="enviar1"
113              value="Eliminar"
            onclick=this.form.action="eliminarParticipante">
115         </div>
          </td>
117       <td colspan="3"><div align="right">
          <input type="submit"
119              name="enviar1"
                 value="Guardar Cambios"
121         onclick=this.form.action="actualizarUsuario">
            </div>
123       </td>
        </s:form>
125
        <a href="<s:url action="personalProyecto"/>">Regresar</a>
127 <!-- ##################################################################### -->
        </div>
129     <div id="right-column">
          <strong class="h">INFO</strong>
131       <div class="box"></div>
        </div>
133 </div>
```

jsp/personal/modificarEliminarUsuario.jsp

```
2 <%@ taglib prefix="s" uri="/struts-tags" %>
  <div id="middle">
4     <div id="left-column">
```

```
      <h3>Header</h3>
      <ul class="nav">
        <li class=""><a href="<s:url action="personalProyecto"/>">
            Personal del Proyecto</a></li>
        <li class=""><a href="<s:url action="participantesProyecto"/>">
            Agregar/Eliminar Personal</a></li>
        <li class=""><a href="<s:url action="modificarEliminarUsuario"/>
            ">Eliminar/Modificar</a></li>
        <li class=""><a href="<s:url action="nuevoUsuario"/>">Nuevo
            Personal</a></li>
      </ul>
      <a href="<s:url action="salirDelSistema"></s:url>" class="link">
          Salir</a>
    </div>

    <div id="center-column">
<!-- ############################################################ -->
    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Personal del Proyecto</h1>
        <div class="breadcrumbs"><a href="#">Personal</a> / <a href="#">
            Personal del Proyecto</a></div>
    </div><br/>

    <p>Proyecto <s:property value="#session.nombreProyecto"/> </p>

    <p>Jefe del Proyecto: <s:property value="proyecto.jefe.
        nombreCompleto"/></p>

    <s:iterator value="proyecto.listaUsuarios">
      <s:property value="nombreCompleto"></s:property><br></br>
      <s:url id="url" action="detallesUsuario">
        <s:param name="idUsuario">
          <s:property value="idUsuario"/>
        </s:param>
      </s:url>
      <s:a href="%{url}">Detalles</s:a><br></br><br></br>
    </s:iterator>

<!-- ############################################################ -->
    </div>
    <div id="right-column">
      <strong class="h">INFO</strong>
      <div class="box"> </div>
    </div>
</div>
```

jsp/personal/personalProyecto.jsp

```

<%@ taglib prefix="s" uri="/struts-tags" %>
<div id="middle">
```

```
    <div id="left-column">
5       <h3>Menu</h3>
        <ul class="nav">
7         <li class=""><a href="<s:url action="personalProyecto"/>">
              Personal del Proyecto</a></li>
          <li class=""><a href="<s:url action="participantesProyecto"/>">
              Agregar/Eliminar Personal</a></li>
9         <li class=""><a href="<s:url action="modificarEliminarUsuario"/>
              ">Eliminar/Modificar</a></li>
          <li class=""><a href="<s:url action="nuevoUsuario"/>">Nuevo
              Personal</a></li>
11      </ul>
    </div>
13
    <div id="center-column">
15<!-- ##################################################################### -->
    <div class="top-bar">
17      <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Personal Actualizado</h1>
19      <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
            Proyectos</a></div>
    </div><br/>
21



23
    El usuario se ha actualizado con los siguientes datos:
25  <p>
    CURP: <s:property value="usuario.idUsuario"/>
27  </p>
    <p>
29  Nombre: <s:property value="usuario.nombre"/>
    </p>
31  <p>
    Apellido Paterno: <s:property value="usuario.ape_paterno"/>
33  </p>
    <p>
35  Apellido Materno: <s:property value="usuario.ape_materno"/>
    </p>
37  <p>
    Correo: <s:property value="usuario.correo"/>
39  </p>
    <p>
41  Telefono: <s:property value="usuario.telefono"/>
    </p>
43  <p>
    Login: <s:property value="usuario.login"/>
45  </p>
    <p>
47  Password: <s:property value="usuario.pass_word"/>
    </p>
49
```

```
51      <p>Roles</p>
        <s:iterator value="usuario.roles" var="it">
53        <s:property value="nombreRol"/><br></br>
        </s:iterator>
55
        <p>
57      <a href="<s:url action="modificarEliminarUsuario"/>">Regresar</a>
        </p>
59  <!-- ############################################################ -->
        </div>
61      <div id="right-column">
          <strong class="h">INFO</strong>
63        <div class="box"> </div>
        </div>
65  </div>
```

jsp/personal/usuarioActualizado.jsp

```
  <%@ taglib prefix="s" uri="/struts-tags" %>
2   <div id="header">
      <a href="index.html" class="logo"><img src="img/logo.gif" width="101
          " height="29" alt="" /></a>
4     <ul id="top-navigation">
        <li><span><span><a href="<s:url action="inicio"/>">Inicio</a></
            span></span></li>
6       <li><span><span><a href="<s:url action="proyectosUsuario"/>">
            Proyectos</a></span></span></li>
        <li class="active"><span><span><a href="<s:url action="
            personalProyecto"/>">Personal</a></span></span></li>
8       <li><span><span><a href="<s:url action="iteraciones"/>">Planificar
            </a></span></span></li>
        <li><span><span><a href="actividadesRup">Actividades</a></span></
            span></li>
10      <li><span><span><a href="rolesSistema">Rol</a></span></span></li>
        <li><span><span><a href="artefactosSistema">Artefactos</a></span
            ></span></li>
12      <li><span><span><a href="tareasUsuario">Tareas</a></span></span></
            li>
        <li><span><span><a href="verPermisos">Seguridad</a></span></span
            ></li>
14
      </ul>
16  </div>
```

jsp/personal/header.jsp

```
2 <%@ taglib prefix="s" uri="/struts-tags" %>

4 <div id="middle">
      <div id="left-column">
6       <h3>Header</h3>
```

```
      <ul class="nav">
        <li class=""><a href="<s:url action="personalProyecto"/>">
            Personal del Proyecto</a></li>
        <li class=""><a href="<s:url action="participantesProyecto"/>">
            Agregar/Eliminar Personal</a></li>
        <li class=""><a href="<s:url action="modificarEliminarUsuario"/>
            ">Eliminar/Modificar</a></li>
        <li class=""><a href="<s:url action="nuevoUsuario"/>">Nuevo
            Personal</a></li>
      </ul>
    </div>

    <div id="center-column">
<!-- ############################################################ -->
    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Detalles del Usuario</h1>
        <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
            Proyectos</a></div>
    </div><br/>
      <p>
      </p>
      Nombre: <s:property value="usuario.nombre"/><br></br>
      Apellido Paterno: <s:property value="usuario.apellido_paterno"/><
          br></br>
      Apellido Materno: <s:property value="usuario.apellido_materno"/><
          br></br>
      Correo: <s:property value="usuario.correo"/><br></br>
      Telefono: <s:property value="usuario.telefono"/><br></br>

      <p>Roles</p>

      <s:iterator value="usuario.roles" var="it">
        Rol: <s:property value="nombreRol"/><br></br>
      </s:iterator>

      <br></br>
      <a href="<s:url action="personalProyecto"/>">Regresar</a>



<!-- ############################################################ -->
    </div>
    <div id="right-column">
      <strong class="h">INFO</strong>
      <div class="box"></div>
    </div>
</div>
```

jsp/personal/detallesUsuario.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>

<div id="middle">
    <div id="left-column">
      <h3>Header</h3>
      <ul class="nav">
        <li class=""><a href="<s:url action="personalProyecto"/>">
            Personal del Proyecto</a></li>
        <li class=""><a href="<s:url action="participantesProyecto"/>">
            Agregar/Eliminar Personal</a></li>
        <li class=""><a href="<s:url action="modificarEliminarUsuario"/>
            ">Eliminar/Modificar</a></li>
        <li class=""><a href="<s:url action="nuevoUsuario"/>">Nuevo
            Personal</a></li>
      </ul>
    </div>

    <div id="center-column">
<!-- ################################################################ -->
    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Personal creado</h1>
        <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
            Proyectos</a></div>
    </div><br/>
      <p>
      El personal con los siguientes datos se ha registrado
          correctamente:
      </p>
      CURP: <s:property value="personal.idPersonal"/><br></br>
      Nombre: <s:property value="personal.nombre"/><br></br>
      Apellido Paterno: <s:property value="personal.apellido_paterno"/><
          br></br>
      Apellido Materno: <s:property value="personal.apellido_materno"/><
          br></br>
      Correo: <s:property value="personal.correo"/><br></br>
      Telefono: <s:property value="personal.telefono"/><br></br>
      Login: <s:property value="personal.login"/><br></br>
      Password: <s:property value="personal.pass_word"/><br></br>

      <a href="<s:url action="obtFormRegistrarPersonal"/>">Regresar</a>



<!-- ################################################################ -->
    </div>
    <div id="right-column">
      <strong class="h">INFO</strong>
```

```
      <div class="box"></div>
    </div>
</div>
```

jsp/personal/personalRegistrado.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>
<div id="middle">
    <div id="left-column">
      <h3>Header</h3>
      <ul class="nav">
        <li class=""><a href="<s:url action="personalProyecto"/>">
            Personal del Proyecto</a></li>
        <li class=""><a href="<s:url action="participantesProyecto"/>">
            Agregar/Eliminar Personal</a></li>
        <li class=""><a href="<s:url action="modificarEliminarUsuario"/>
            ">Eliminar/Modificar</a></li>
        <li class=""><a href="<s:url action="nuevoUsuario"/>">Nuevo
            Personal</a></li>
      </ul>
      <a href="<s:url action="salirDelSistema"></s:url>" class="link">
          Salir</a>
    </div>

    <div id="center-column">
<!-- ##################################################### -->
    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Personal del Proyecto</h1>
        <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
            Proyectos</a></div>
    </div><br/>

    <p>Personal del Proyecto <s:property value="#session.nombreProyecto"
       /> </p>
    <p>Jefe del Proyecto: <s:property value="proyecto.jefe.
        nombreCompleto"/></p>

    <p>Lista de Participantes del Proyecto</p>
    <s:iterator value="proyecto.listaUsuarios">
      <s:property value="nombreCompleto"></s:property><br></br>
    </s:iterator>

    <a href="<s:url action="participantesProyecto"/>">Regresar</a>

<!-- ##################################################### -->
    </div>
    <div id="right-column">
      <strong class="h">INFO</strong>
      <div class="box"></div>
    </div>
```

```
</div>
```

jsp/personal/personalProyectoActualizada.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>
<div id="middle">
    <div id="left-column">
      <h3>Menu</h3>
      <ul class="nav">
      </ul>
    </div>

    <div id="center-column">
<!-- ############################################################ -->
    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Seguridad</h1>
        <div class="breadcrumbs"><a href="#">Seguridad</a> / <a href="#"
            >Modificar Accion</a></div>
    </div><br/>
    <p>Modificar la accion</p>
      <s:form>
        <s:hidden name="idAccion"></s:hidden>
        <s:hidden name="nuevaccion.id"></s:hidden>
        <s:textfield name="nuevaAccion.nombre" label="Accion"></s:
            textfield>
        <s:textarea name="nuevaAccion.descripcion" label="Descripcion"
            ></s:textarea>
        <s:textfield name="nuevaAccion.categoria" label="Categoria"></s:
            textfield>
        <td colspan="3"><div align="right">
        <input type="submit"
              name="enviar1"
              value="Eliminar"
          onclick=this.form.action="eliminarAccion">
          </div>
      </td>
      <td colspan="3"><div align="right">
        <input type="submit"
              name="enviar1"
              value="Guardar Cambios"
          onclick=this.form.action="actualizarAccion">
          </div>
      </td>
      </s:form >
<!-- ############################################################ -->
    </div>
    <div id="right-column">
      <strong class="h">INFO</strong>
      <div class="box"></div>
    </div>
```

```
45 </div>
```

jsp/seguridad/modificarEliminarAccion.jsp

```
1
<%@ taglib prefix="s" uri="/struts-tags" %>
3 <div id="middle">
    <div id="left-column">
5     <h3>Menu</h3>
      <ul class="nav">
7       <li><a href="nuevaAccion">Agregar Accion</a></li>
      </ul>
9   </div>

11  <div id="center-column">
<!-- ############################################################ -->
13  <div class="top-bar">
      <a href="<s:url action="salirDelSistema"></s:url>" class="button
          ">Salir </a>
15    <h1>Permisos</h1>
      <div class="breadcrumbs"><a href="#">Seguridad</a> / <a href="#"
         >Permisos</a></div>
17  </div><br/>
    <p>Seleccione un rol</p>
19  <s:form action="verPermisos">
      <s:select name="rolActual" list="roles"
21      listKey="nombreRol" listValue="nombreRol"></s:select>
      <s:submit value="Aceptar"></s:submit>
23  </s:form>

25

    <p></p>
27
    <table class="tareas">
29    <caption>Permisos</caption>
      <thead>
31      <tr>
          <th>IdAccion</th>
33        <th>Accion</th>
          <th>Descripcion</th>
35        <th>Categoria</th>
          <th>Rol</th>
37        <th>Permitido</th>
          <th>Cambiar</th>
39      </tr>
      </thead>
41    <tbody>
        <s:iterator value="permisos" var="it2">
43        <s:url id="mod" action="modificarEliminarAccion">
            <s:param name="idAccion">
45            <s:property value="idAccion"/>
            </s:param>
47        </s:url>
```

```
49          <s:url id="dt2" action="guardarCambiosPermisos">
              <s:param name="idAccion">
51              <s:property value="idAccion"/>
              </s:param>
53            <s:param name="rolAccion">
                <s:property value="rol"/>
55            </s:param>
            </s:url>
57          <tr>
              <td><s:property value="idAccion"/></td>
59            <td><s:a href="%{mod}"> <s:property value="accion"/> </s:a
                  ></td>
              <td><s:property value="descripcionAccion"/></td>
61            <td><s:property value="categoria"/></td>
              <td><s:property value="rol"/></td>
63            <td><s:property value="permitidoS"/> </td>
              <td>
65              <s:a href="%{dt2}">Cambiar </s:a><br></br>
              </td>
67          </tr>
          </s:iterator>
69

        </tbody>
71    </table>
<!-- ################################################################ -->
73    </div>
      <div id="right-column">
75      <strong class="h">INFO</strong>
        <div class="box"></div>
77    </div>
</div>
```

jsp/seguridad/permisos.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>
2   <div id="header">
      <a href="index.html" class="logo"><img src="img/logo.gif" width="101
          " height="29" alt="" /></a>
4     Proyecto: <s:property value="#session.nombreProyecto"/><br></br>
      Usuario: <s:property value="#session.nombreUsuario"/><br></br>
6     <ul id="top-navigation">
        <li><span><span><a href="<s:url action="inicio"/>">Inicio</a></
            span></span></li>
8       <li ><span><span><a href="<s:url action="proyectosUsuario"/>">
            Proyecto</a></span></span></li>
        <li><span><span><a href="<s:url action="personalProyecto"/>">
            Personal</a></span></span></li>
10      <li><span><span><a href="<s:url action="iteraciones"/>">Planificar
            </a></span></span></li>
        <li><span><span><a href="actividadesRup">Actividades</a></span></
            span></li>
12      <li><span><span><a href="rolesSistema">Rol</a></span></span></li>
```

```
      <li><span><span><a href="artefactosSistema">Artefactos</a></span
          ></span></li>
      <li><span><span><a href="tareasUsuario">Tareas</a></span></span></
          li>
      <li class="active"><span><span><a href="verPermisos">Seguridad</a
          ></span></span></li>
    </ul>
  </div>
```

jsp/seguridad/header.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>
<div id="middle">
    <div id="left-column">
      <h3>Menu</h3>
      <ul class="nav">
      </ul>
    </div>

    <div id="center-column">
<!-- ############################################################ -->
    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Seguridad</h1>
        <div class="breadcrumbs"><a href="#">Seguridad</a> / <a href="#"
            >Nueva Accion</a></div>
    </div><br/>

    <p>Introduzca los datos</p>

      <s:form action="registrarAccion">
        <s:textfield name="nuevaAccion.nombre" label="Accion"></s:
            textfield>
        <s:textarea name="nuevaAccion.descripcion" label="Descripcion"
            ></s:textarea>
        <s:textfield name="nuevaAccion.urlDescripcion" label="Categoria"
            ></s:textfield>
        <s:submit value="Registrar"></s:submit>
      </s:form >
<!-- ############################################################ -->
    </div>
    <div id="right-column">
      <strong class="h">INFO</strong>
      <div class="box"></div>
    </div>
</div>
```

jsp/seguridad/nuevaAccion.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>
<div id="middle">
```

```
    <s:div id="left-column">
4           <h3>Menu</h3>
            <ul id="menuSecundario" class="nav">
6             <li class=""><a href="<s:url action="inicio"/>">Bienvenida</
                  a></li>
              <li class=""><a href="<s:url action="login"/>">Entrar</a></
                  li>
8           </ul>
            <!--
10          <a href="#" class="link">Entrar</a>
             -->
12          <a href="<s:url action="salirDelSistema"></s:url>" class="link
                ">Salir</a>
    </s:div>
14
    <div id="center-column">
16<!-- ############################################################ -->
        <div class="top-bar">
18        <a href="<s:url action="salirDelSistema"></s:url>" class="
              button">Salir</a>
          <h1>No autorizado</h1>
20
          <div class="breadcrumbs"><a href="<s:url action="inicio"></s:
              url>">Inicio</a> / <a href="#">Bienvenida</a></div>
22      </div><br/>

24      <p class="errorMessage">
          <h3>No tiene los permisos para ejecutar al accion</h3>
26      </p>
        <p>
28        <s:actionmessage/>
          <s:actionerror/>
30        <s:actionerror/>

32      </p>


34
<!-- ############################################################ -->
36  </div>
    <div id="right-column">
38    <strong class="h">INFO</strong>
      <div class="box"> </div>
40  </div>
</div>
```

jsp/sistema/sinPermisos.jsp

```
1<%@ taglib prefix="s" uri="/struts-tags" %>
  <div id="header">
3   <a href="index.html" class="logo"><img src="img/logo.gif" width="101
        " height="29" alt="" /></a>
    <ul id="top-navigation">
```

```
5       <li class="active"><span><span><a href="<s:url action="inicio"/>">
            Inicio</a></span></span></li>
        <li><span><span><a href="<s:url action="proyectosUsuario"/>">
            Proyecto</a></span></span></li>
7       <li><span><span><a href="<s:url action="personalProyecto"/>">
            Personal</a></span></span></li>
        <li><span><span><a href="<s:url action="iteraciones"/>">Planificar
            </a></span></span></li>
9       <li><span><span><a href="actividadesRup">Actividades</a></span></
            span></li>
        <li><span><span><a href="rolesSistema">Rol</a></span></span></li>
11      <li><span><span><a href="artefactosSistema">Artefactos</a></span
            ></span></li>
        <li><span><span><a href="tareasUsuario">Tareas</a></span></span></
            li>
13      <li><span><span><a href="verPermisos">Seguridad</a></span></span
            ></li>
    </ul>
15  </div>
```

jsp/sistema/header.jsp

```
1<%@ taglib prefix="s" uri="/struts-tags" %>
<div id="middle">
3    <s:div id="left-column">
          <h3>Menu</h3>
5         <ul id="menuSecundario" class="nav">
            <li class=""><a href="<s:url action="inicio"/>">Bienvenida</
                a></li>
7           <li class=""><a href="<s:url action="login"/>">Entrar</a></
                li>
          </ul>
9         <!--
          <a href="#" class="link">Entrar</a>
11         -->
          <a href="#" class="link">Salir</a>
13   </s:div>

15   <div id="center-column">
<!-- ############################################################ -->
17       <div class="top-bar">
           <a href="<s:url action="salir"></s:url>" class="button">Salir
               </a>
19         <h1>Error</h1>

21         <div class="breadcrumbs"><a href="<s:url action="inicio"></s:
               url>">Inicio</a> / <a href="#">Bienvenida</a></div>
         </div><br/>
23
         <p class="errorMessage">
25         <h3>Error</h3>
         </p>
27       <p>
```

```
            <s:actionmessage/>
29          <s:actionerror/>
            <s:actionerror/>
31
          </p>
33

35 <!-- ############################################################# -->
      </div>
37    <div id="right-column">
        <strong class="h">INFO</strong>
39      <div class="box">Detect and eliminate viruses and Trojan horses,
            even new and unknown ones. Detect and eliminate viruses and
            Trojan horses, even new and </div>
      </div>
41 </div>
```

jsp/sistema/error.jsp

```
1 <%@ taglib prefix="s" uri="/struts-tags" %>
  <div id="middle">
3     <s:div id="left-column">
            <h3>Menu</h3>
5           <ul id="menuSecundario" class="nav">
              <li class=""><a href="<s:url action="inicio"/>">Bienvenida</
                  a></li>
7             <li class=""><a href="<s:url action="login"/>">Entrar</a></
                  li>
            </ul>
9           <!--
            <a href="#" class="link">Entrar</a>
11           -->
            <a href="<s:url action="salirDelSistema"></s:url>" class="link
                ">Salir</a>
13    </s:div>

15    <div id="center-column">
  <!-- ############################################################# -->
17        <div class="top-bar">
            <a href="<s:url action="salirDelSistema"></s:url>" class="
                button">Salir</a>
19          <h1>Bienvenida</h1>

21          <div class="breadcrumbs"><a href="<s:url action="inicio"></s:
                url>">Inicio</a> / <a href="#">Bienvenida</a></div>
          </div><br/>
23        <p>
          Bienvenido a al Sistema Gestor de Proyectos (SGPv1.0).
25        </p>

27



29
```

```
<!-- ############################################################ -->
    </div>
    <div id="right-column">
      <strong class="h">INFO</strong>
      <div class="box">Detect and eliminate viruses and Trojan horses,
          even new and unknown ones. Detect and eliminate viruses and
          Trojan horses, even new and </div>
    </div>
</div>
```

jsp/sistema/inicio.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>

<link type="text/css" href="/PrototipoWeb/jsp/css/jquery-ui-1.8.2.custom
    .css" rel="Stylesheet" />
<script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery-1.4.2.
    min.js"></script>
<script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery-ui
    -1.8.2.custom.min.js"></script>
<script type="text/javascript" src="/PrototipoWeb/jsp/js/md5-min.js"></
    script>

<script type="text/javascript">
<!--
var x= $(document);
x.ready(inicializarEventos);

function inicializarEventos(){
  var x=$("#user_id");
  x.change(cifrar);
  x = $("#password");
  x.attr("value",null);
}

function cifrar(){
  var x=$("#user_id");
  var p = hex_md5(x.attr("value"));
  x = $("#password");
  x.attr("value",p);
}
//-->
</script>


<div id="middle">
    <s:div id="left-column">
          <h3>Menu</h3>
          <ul id="menuSecundario" class="nav">
            <li class=""><a href="<s:url action="inicio"/>">Bienvenida</
                a></li>
```

```
            <li class=""><a href="<s:url action="login"/>">Entrar</a></
                li>
37          </ul>
            <!--
39          <a href="#" class="link">Entrar</a>
             -->
41          <a href="<s:url action="salirDelSistema"></s:url>" class="link
                ">Salir</a>
    </s:div>
43
    <div id="center-column">
45<!-- ############################################################ -->
      <div class="top-bar">
47          <a  class="button" href="<s:url action="salirDelSistema"/>">
                Salir</a>
            <h1>Login</h1>
49          <div class="breadcrumbs"><a href="#">Inicio</a> / <a href="#">
                Entrar</a></div>
      </div><br/>
51    <s:if test="#session.usuario!=null">
        Usted se ha autenticado como:
53      <p>
        <s:property value="#session.nombreUsuario"/>
55      </p>
      </s:if><s:else>
57


59      <p>Introduzca sus datos</p>
        <div class="errorMessage">
61        <s:actionmessage/>
        </div>
63      <s:form action="validarUsuario">
          <s:textfield id="login" name="login" label="Login (correo)"></
              s:textfield>
65        <s:password id="user_id" label="Password"></s:password>
          <s:hidden id="password" name="password"></s:hidden>
67        <s:submit value="Validar"></s:submit>
        </s:form >
69      </s:else>

71<!-- ############################################################ -->
    </div>
73  <div id="right-column">
      <strong class="h">INFO</strong>
75    <div class="box">Detect and eliminate viruses and Trojan horses,
          even new and unknown ones. Detect and eliminate viruses and
          Trojan horses, even new and </div>
    </div>
77</div>
```

jsp/sistema/login.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>

<div id="middle">
    <div id="left-column">
      <h3>Menu</h3>
      <ul class="nav">
        <li><a href="<s:url action="nuevaIteracion"/>">Nueva Iteracion</
            a></li>
        <li><a href="<s:url action="proyectosUsuario"/>">Cambiar de
            Proyecto Actual</a></li>
      </ul>
    </div>

    <div id="center-column">
<!-- ##################################################################### -->
    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Tarea Eliminada</h1>
        <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
            Proyectos</a></div>
    </div><br/>
    <p>
      La tarea se ha eliminado...
    </p>


    <a href="<s:url action="tareas">
      <s:param name="numIteracion" value="%{numIteracion}" />
    </s:url>">Regresar</a>

<!-- ##################################################################### -->
    </div>
    <div id="right-column">
      <strong class="h">INFO</strong>
      <div class="box"> </div>
    </div>
</div>
```

jsp/planificar/tereaEliminada.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>
<link type="text/css" href="/PrototipoWeb/jsp/css/jquery-ui-1.8.2.custom
    .css" rel="Stylesheet" />
<script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery-1.4.2.
    min.js"></script>
<script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery-ui
    -1.8.2.custom.min.js"></script>
<script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery.ui.
    datepicker-es.js"></script>
<script type="text/javascript" src="/PrototipoWeb/jsp/js/
    funcionesNuevaTarea.js"></script>
```

```
8
<div id="middle">
10      <div id="left-column">
          <h3>Menu</h3>
12        <ul class="nav">
            <li><a href="<s:url action="nuevaIteracion"/>">Nueva Iteracion</
                a></li>
14          <li><a href="<s:url action="proyectosUsuario"/>">Cambiar de
                Proyecto Actual</a></li>
          </ul>
16      </div>

18      <div id="center-column">
<!-- ############################################################ -->
20      <div class="top-bar">
            <a href="<s:url action="salirDelSistema"></s:url>" class="button
                ">Salir </a>
22          <h1>Nueva Tarea</h1>
            <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
                Proyectos</a></div>
24      </div><br/>
        <p>
26        Proporcione los datos para la nueva tarea:
        </p>
28      <s:form action="registrarTarea">
          <s:hidden id="numIteracion" name="numIteracion"></s:hidden>
30        <s:textfield name="tarea.nombreTarea" label="Nombre de la Tarea: "
              ></s:textfield>
          <s:textfield id="fechaInicio" name="tarea.fechaInicio" label="
              Fecha de Inicio"></s:textfield>
32        <s:textfield id="fechaFin" name="tarea.fechaFin" label="Fecha de
              Termino"></s:textfield>
          <s:textfield name="tarea.porcentaje" label="Completado"></s:
              textfield>
34
          <s:select id="selDisciplina" name="disciplinaElegida" label="
              Disciplina" list="disciplinas"
36          listValue="nombre" listKey="idDisciplina"
            size="5"
38          headerKey="0"
            headerValue="<---------Seleccione una Disciplina--------->"
40          onchange="actualizarActividades();">
          </s:select>
42
          <s:select id="selActividad" name="actividadElegida" label="
              Actividad" list="{}" listKey="idActividad"
44          headerKey="0"
            headerValue="<---------Seleccione una Actividad--------->"
46           listValue="nombre" size="5"
             onchange="actualizarRecursos();">
48        </s:select>
```

```
50        <s:select id="selPadre" name="padreElegido" label="Tarea Padre"
              list="{}"
            listValue="nombreTarea" listKey="numTarea"
52          headerKey="0"
            headerValue="<----------Tarea Padre------>"
54          onchange="actualizarSubtareas();"
            size="5"></s:select>
56
        <s:optiontransferselect
58          id="predecesoresElegidos"
            label="Predecesores"
60          name="tareasPredecesoras"
                leftTitle="Tareas Predecesoras"
62              rightTitle="Tareas"
                list="{}"
64              listKey="idTarea"
                listValue="nombreTarea"
66              multiple="true"
                headerKey="0"
68              headerValue="---------- Tareas predecesoras ----------"
                size="6"
70              doubleId="predecesoresCandidatos"
                doubleList="{}"
72              doubleListKey="numTarea"
                doubleListValue="nombreTarea"
74              doubleName=""
                doubleHeaderKey="0"
76              doubleHeaderValue="<---------- Tareas ------>"
                doubleMultiple="true"
78              doubleSize="6"
                allowUpDownOnLeft="false"
80              allowUpDownOnRight="false"
                allowAddAllToLeft="false"
82              allowAddAllToRight="false"
                allowSelectAll="false"
84      />

86      <s:optiontransferselect id="selResponsables"
          name="responsables"
88              leftTitle="Personal Asignado"
                rightTitle="Personal Disponible"
90              list="{}"
                listKey="idUsuario"
92              listValue="nombreCompleto"
                multiple="true"
94              headerKey="0"
                headerValue="<---------- Ningun Usuario ------>"
96              size="12"
                emptyOption="false"
98              doubleId="selResponsablesCandidatos"
                doubleList="{}"
100             doubleListKey="idUsuario"
```

```
                doubleListValue="nombreCompleto"
102             doubleName="noresponsables"
                doubleHeaderKey="0"
104             doubleHeaderValue="<---------- Usuarios ---------->"
                doubleMultiple="true"
106             doubleSize="12"
                allowUpDownOnLeft="false"
108             allowUpDownOnRight="false"
                allowAddAllToLeft="false"
110             allowAddAllToRight="false"
                allowSelectAll="false"
112         />

114         <s:submit value=" Aceptar "></s:submit>
        </s:form>
116
<!-- ############################################################ -->
118     </div>
        <div id="right-column">
120       <strong class="h">INFO</strong>
          <div class="box"></div>
122     </div>
</div>
```

jsp/planificar/nuevaTarea.jsp

```
1 <%@ taglib prefix="s" uri="/struts-tags" %>
  <link type="text/css" href="/PrototipoWeb/jsp/css/jquery-ui-1.8.2.custom
      .css" rel="Stylesheet" />
3 <script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery-1.4.2.
      min.js"></script>
  <script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery-ui
      -1.8.2.custom.min.js"></script>
5 <script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery.ui.
      datepicker-es.js"></script>
  <script type="text/javascript" src="/PrototipoWeb/jsp/js/
      funcionesModificarTarea.js"></script>
7 <div id="middle">
      <div id="left-column">
9       <h3>Menu</h3>
        <ul class="nav">
11        <li><a href="<s:url action="nuevaIteracion"/>">Nueva Iteracion</
              a></li>
          <li><a href="<s:url action="proyectosUsuario"/>">Cambiar
              Proyecto Actual</a></li>
13      </ul>
      </div>
15
      <div id="center-column">
17 <!-- ############################################################ -->
      <div class="top-bar">
19        <a href="<s:url action="salirDelSistema"></s:url>" class="button
              ">Salir </a>
```

```
        <h1>Modificar Tarea</h1>
21      <div class="breadcrumbs"><a href="#">Planificar</a> / <a href="#
            ">Iteraciones</a> / <a href="#">Modificar Tarea</a> </div>
    </div><br/>
23  <p>
      Proyecto Actual: ${idProyecto}
25    Iteracion: ${numIteracion}
      Tarea: ${numTarea}
27  </p>
    <s:form>
29    <s:hidden name="idProyecto"></s:hidden>
      <s:hidden name="numIteracion"></s:hidden>
31    <s:hidden name="numTarea"></s:hidden>

33    <s:textfield name="tarea.nombreTarea" label="Nombre de la Tarea: "
          ></s:textfield>
      <s:textfield id="fechaInicio" name="tarea.fechaInicio"
35     label="Fecha de Inicio"></s:textfield>

37    Fecha: <s:property value="format2"></s:property>
      <s:textfield id="fechaFin" name="tarea.fechaFin" label="Fecha de
          Termino"></s:textfield>
39    <s:textfield name="tarea.porcentaje" label="Completado"></s:
          textfield>

41    <s:hidden name="disciplinaElegida"></s:hidden>
      <tr>
43    <td><p>Disciplina</p></td>
      <td colspan="2"><div align="left">
45      <s:iterator value="disciplinas" var="it">
        <s:if test="idDisciplina==tarea.actividad.disciplina.
            idDisciplina">
47        Disciplina: <s:property value="nombre"></s:property>
        </s:if>
49      </s:iterator>
      </div>
51    </td>
      </tr>
53
      <tr>
55    <td><p>Actividad</p></td>
      <td colspan="2"><div align="left">
57    <select id="actividadElegida" size="1" name="actividadElegida"
          onchange="actualizarRecursos();">
        <s:iterator value="actividades" var="it">
59      <s:if test="idActividad==tarea.actividad.idActividad">
          <option value='<s:property value="idActividad"></s:property>'
              selected><s:property value="nombre"></s:property></option>
61      </s:if><s:else>
          <option value='<s:property value="idActividad"></s:property>'
              ><s:property value="nombre"></s:property></option>
63      </s:else>
```

```
        </s:iterator>
65      </select>
        </div>
67      </td>
        </tr>
69
        <tr>
71      <td><p>Tarea Padre</p></td>
        <td colspan="2"><div align="left">
73      <select id="padreElegido" size="1" name="padreElegido">
          <s:iterator value="padresCandidatos" var="it">
75        <s:if test="idTarea.idCadena==tarea.idTarea.idCadena">
            <option value='<s:property value="idTarea.idCadena"></s:
                property>' selected><s:property value="nombreTarea"></s:
                property></option>
77        </s:if><s:else>
            <option value='<s:property value="idTarea.idCadena"></s:
                property>'><s:property value="nombreTarea"></s:property></
                option>
79        </s:else>
          </s:iterator>
81      </select>
        </div>
83      </td>
        </tr>
85

87      <s:optiontransferselect
          id="predecesoresElegidos"
89        label="Predecesores"
          name="predecesoresElegidos"
91            leftTitle="Tareas Predecesoras"
              rightTitle="Tareas"
93            list="tarea.predecesores"
              listKey="idTarea.idCadena"
95            listValue="nombreTarea"
              multiple="true"
97            headerKey="0"
              headerValue="--------------------------"
99            size="6"
              doubleId="predecesoresCandidatos"
101           doubleList="prececesoresCandidatos"
              doubleListKey="idTarea.idCadena"
103           doubleListValue="nombreTarea"
              doubleName=""
105           doubleHeaderKey="0"
              doubleMultiple="true"
107           doubleSize="6"
              allowUpDownOnLeft="false"
109           allowUpDownOnRight="false"
              allowAddAllToLeft="false"
111           allowAddAllToRight="false"
```

```
                allowSelectAll="false"
113         />

115         <s:optiontransferselect id="recursosAsignado" label="Recursos"
              name="recursosAsignado"
117               leftTitle="Personal Asignado"
                  rightTitle="Personal Disponible"
119               list="tarea.recursos"
                  listKey="idUsuario"
121               listValue="nombreCompleto"
                  multiple="true"
123               headerKey="headerKey"
                  headerValue="--------------------------------"
125               size="6"
                  emptyOption="false"
127               doubleId="recursosCandidatos"
                  doubleList="recursosCandidatos"
129               doubleListKey="idUsuario"
                  doubleListValue="nombreCompleto"
131               doubleName=""
                  doubleHeaderKey="doubleHeaderKey"
133               doubleMultiple="true"
                  doubleSize="6"
135               allowUpDownOnLeft="false"
                  allowUpDownOnRight="false"
137               allowAddAllToLeft="false"
                  allowAddAllToRight="false"
139               allowSelectAll="false"
          />
141       <tr>
          <td colspan="1"><div align="right">
143         <input type="submit"
                 name="enviar1"
145              value="Eliminar Tarea"
            onclick=this.form.action="eliminarTarea">
147         </div>
          </td>
149
          <td colspan="2"><div align="right">
151         <input type="submit"
                 name="enviar1"
153              value="Guardar Cambios"
            onclick=this.form.action="guardarTareaModificada">
155         </div>
          </td>
157       </tr>
        </s:form>
159
<!-- ########################################################## -->
161     </div>
        <div id="right-column">
163       <strong class="h">INFO</strong>
```

```
      <div class="box"> </div>
    </div>
</div>
```

jsp/planificar/modificarTarea.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8" %>
<%@ taglib prefix="s" uri="/struts-tags" %>
<%@ taglib uri="http://tiles.apache.org/tags-tiles" prefix="tiles"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://
    www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title><tiles:getAsString name="pageTitle"/></title>
    <link href="<s:url value="/jsp/css/admin2.css"/>" rel="stylesheet"
          type="text/css"/>
</head>
<body>
Proyecto: <s:property value="#session.nombreProyecto"/><br></br>
Usuario: <s:property value="#session.nombreUsuario"/><br></br>
<div id="main">
<s:property value="#session.nombreProyecto"/>
  <!-- header -->
  <tiles:insertAttribute name="header"/>
  <!-- middle -->
  <tiles:insertAttribute name="middle"/>

  <!-- footer -->
  <tiles:insertAttribute name="footer"/>
</div>

</body>
</html>
```

jsp/planificar/tareas/estructuraGeneral.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>

<div id="middle">
    <div id="left-column">
      <h3>Menu</h3>
      <ul class="nav">
        <li><a href="<s:url action="nuevaIteracion"/>">Nueva Iteracion</
            a></li>
        <li><a href="<s:url action="proyectosUsuario"/>">Cambiar de
            Proyecto Actual</a></li>
      </ul>
    </div>

    <div id="center-column">
<!-- ######################################################### -->
```

```
14      <div class="top-bar">
            <a href="<s:url action="salirDelSistema"></s:url>" class="button
                ">Salir </a>
16          <h1>Iteraciones del Proyecto</h1>
            <div class="breadcrumbs"><a href="#">Planificar</a> / <a href="#
                ">Iteraciones del Proyecto</a></div>
18      </div><br/>
        <br></br>
20      <br></br>
            <br></br>
22      <br></br>
        <div>
24      <p>
          Las iteraciones del proyecto son:
26      </p>
        </div>
28
        <s:iterator value="iteraciones" var="it">
30        <h3> Nombre: <s:property value="nombre"/><br></br></h3>
          Descripcion: <s:property value="descripcion"/>
32        <br></br>


34
          <!-- Agregamos una liga para que el usuario
36          vea las tareas de la iteracion -->
          <s:url id="dt0" action="tareas">
38            <s:param name="numIteracion">
                <s:property value="id.numIteracion"/>
40            </s:param>
          </s:url>
42
          <!-- Agregamos una liga para que el usuario
44          pueda modificar/Eliminar la iteracion -->
          <s:url id="dt1" action="modificarEliminarIteracion">
46            <s:param name="numIteracion">
                <s:property value="id.numIteracion"/>
48            </s:param>
          </s:url>
50
          <s:a href="%{dt0}">Ver Tareas</s:a><br></br>
52        <s:a href="%{dt1}">Eliminar/Modificar</s:a><br></br><br></br>


54
        </s:iterator>
56
<!-- ############################################################ -->
58      </div>
        <div id="right-column">
60        <strong class="h">INFO</strong>
          <div class="box"></div>
62      </div>
</div>
```

jsp/planificar/iteraciones.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>

<div id="middle">
    <div id="left-column">
      <h3>Menu</h3>
      <ul class="nav">
        <li><a href="<s:url action="nuevaIteracion"/>">Nueva Iteracion</
            a></li>
        <li><a href="<s:url action="datosProyectos"/>">Eliminar</a></li>
        <li><a href="<s:url action="listarProyectos"/>">Cambiar Proyecto
             Actual</a></li>
      </ul>
      <a href="<s:url action="salirDelSistema"></s:url>" class="link">
          Salir</a>
      <a href="#" class="link">Link here</a>
    </div>

    <div id="center-column">
<!-- ######################################################### -->
    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Proyectos</h1>
        <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
            Proyectos</a></div>
    </div><br/>
    <p>
      La tarea se ha eliminado...
    </p>


    <a href="<s:url action="tareas">
      <s:param name="numIteracion" value="%{numIteracion}" />
    </s:url>">Regresar</a>

<!-- ######################################################### -->
    </div>
    <div id="right-column">
      <strong class="h">INFO</strong>
      <div class="box"> </div>
    </div>
</div>
```

jsp/planificar/tareaEliminada1.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>
<div id="middle">
    <div id="left-column">
      <h3>Menu</h3>
```

```
      <ul class="nav">
        <li><a href="<s:url action="nuevaIteracion"/>">Nueva Iteracion</
            a></li>
        <li><a href="<s:url action="proyectosUsuario"/>">Cambiar de
            Proyecto Actual</a></li>
      </ul>
    </div>

    <div id="center-column">
<!-- ########################################################## -->
    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Tarea Registrada</h1>
        <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
            Proyectos</a></div>
    </div><br/>
    <p>
      Tarea Agregada Exitosamente
    </p>
    <s:hidden id="numIteracion" name="numIteracion"></s:hidden>

      <s:url id="dt1" action="tareas">
          <s:param name="numIteracion">
            <s:property value="numIteracion"/>
          </s:param>
      </s:url>

    <s:a href="%{dt1}">Regresar</s:a>

<!-- ########################################################## -->
    </div>
    <div id="right-column">
      <strong class="h">INFO</strong>
      <div class="box"></div>
    </div>
</div>
```

jsp/planificar/tareaRegistrada.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>
    <link href="<s:url value="/jsp/css/jsgantt.css"/>" rel="stylesheet"
          type="text/css"/>
<script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery-1.4.2.js
    "></script>
<script type="text/javascript" src="/PrototipoWeb/jsp/js/jsgantt.js"></
    script>
<script type="text/javascript" src="/PrototipoWeb/jsp/js/funcionesTareas
    .js"></script>

<div id="middle">
<!-- ########################################################## -->
```

```
<div id="center-column2">
    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Tareas de la Iteracion</h1>
        <div class="breadcrumbs"><a href="iteraciones">Planificar</a> /
            <a href="#">Ver Tareas</a></div>
    </div>


  <s:hidden id="numIteracion" name="numIteracion"></s:hidden>


  <a href="<s:url action="nuevaTarea">
    <s:param name="numIteracion" value="%{numIteracion}" />
  </s:url>">Nueva Tarea</a>

  <p>Tareas del la Iteracion</p>
  <div id="cargando" style="display:none; color: green;">
    <IMG SRC="/PrototipoWeb/jsp/img/jquery/ajax-loader.gif">
  </div>


  <div style="position:relative" class="gantt" id="GanttChartDIV">
  <script language="javascript">
  g = new JSGantt.GanttChart('g',document.getElementById('GanttChartDIV'
      ), 'day');
  personalizar();
  fetchJSONData(document.getElementById('numIteracion').value);
  </script>
  </div>
  <a href="iteraciones">Regresar</a>
</div>


</div>
```

jsp/planificar/tareas.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>

<div id="middle">
    <div id="left-column">
      <h3>Menu</h3>
      <ul class="nav">
        <li><a href="<s:url action="nuevaIteracion"/>">Nueva Iteracion</
            a></li>
        <li><a href="<s:url action="proyectosUsuario"/>">Cambiar de
            Proyecto Actual</a></li>
      </ul>
    </div>

    <div id="center-column">
```

```
<!-- ############################################################ -->
    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Tarea Modificada</h1>
        <div class="breadcrumbs"><a href="#">Planificar</a> / <a href="#
            ">Iteraciones</a></div>
    </div><br/>
    <p>
    ------------------------------------------------------------------
    </p>
    <p>
      La tarea se guardo correctamente.
    </p>


    <a href="<s:url action="tareas">
      <s:param name="numIteracion" value="%{numIteracion}" />
    </s:url>">Regresar</a>

<!-- ############################################################ -->
    </div>
    <div id="right-column">
      <strong class="h">INFO</strong>
      <div class="box"></div>
    </div>
</div>
```

jsp/planificar/tareaModificada.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>
<link type="text/css" href="/PrototipoWeb/jsp/css/jquery-ui-1.8.2.custom
    .css" rel="Stylesheet" />
<script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery-1.4.2.
    min.js"></script>
<script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery-ui
    -1.8.2.custom.min.js"></script>
<script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery.ui.
    datepicker-es.js"></script>
<script type="text/javascript">
<!--
var x = $(document);
x.ready(inicializarEventos);

function inicializarEventos(){
  var x = $("#fechaInicio");
  x.datepicker();
}
//-->
</script>


```

```
<div id="middle">
    <div id="left-column">
      <h3>Menu</h3>
      <ul class="nav">
        <li><a href="<s:url action="nuevaIteracion"/>">Nueva Iteracion</
            a></li>
        <li><a href="<s:url action="proyectosUsuario"/>">Cambiar de
            Proyecto Actual</a></li>
      </ul>
    </div>

    <div id="center-column">
<!-- ######################################################## -->
    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Modificar/Eliminar Iteracion</h1>
        <div class="breadcrumbs"><a href="#">Planificar</a> / <a href="#
            ">Modificar/Eliminar Iteracion</a></div>
    </div><br/>
    <p>
      Proporciones los datos correspondientes.
    </p>

    <s:form>
      <s:hidden name="iteracion.id.numIteracion"></s:hidden>
      <s:hidden name="iteracion.id.idProyecto"></s:hidden>
      <s:textfield name="iteracion.nombre" label="Nombre"></s:textfield>
      <s:textarea name="iteracion.descripcion" label="Descripcion" cols=
          "30" rows="8"></s:textarea>


      <tr>
      <td><p>Fase</p></td>
      <td colspan="2"><div align="left">
      <select name="faseElegida">
      <s:iterator value="fases" var="it">
        <s:if test="iteracion.fase.idFase==idFase">
          <option value='<s:property value="idFase"></s:property>'
              selected><s:property value="nombre"></s:property></option>
        </s:if><s:else>
          <option value='<s:property value="idFase"></s:property>'><s:
              property value="nombre"></s:property></option>
        </s:else>
      </s:iterator>
      </select>
      </div>
      </td>
      </tr>

      <s:textfield id="fechaInicio" name="iteracion.fechaInicio" label="
          Fecha Inicio"></s:textfield>
```

```
65        <td colspan="1"><div align="left"></div>
            <input type="submit"
67                name="enviar2"
                  value="Eliminar"
69              onclick=this.form.action="eliminarIteracion">
          </td>
71          <td colspan="3"><div align="right">
            <input type="submit"
73                name="enviar2"
                  value="Guardar"
75              onclick=this.form.action="guardarCambiosIteracion">
              </div>
77        </td>
      </s:form>
79
<!-- ############################################################ -->
81    </div>
      <div id="right-column">
83      <strong class="h">INFO</strong>
        <div class="box"></div>
85    </div>
</div>
```

jsp/planificar/modificarEliminarIteracion.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>
2
<div id="middle">
4     <div id="left-column">
        <h3>Menu</h3>
6       <ul class="nav">
          <li><a href="<s:url action="obtFormProyecto"/>">Nuevo</a></li>
8         <li><a href="<s:url action="datosProyectos"/>">Eliminar</a></li>
          <li><a href="<s:url action="listarProyectos"/>">Cambiar Proyecto
              Actual</a></li>
10      </ul>
        <a href="<s:url action="salirDelSistema"></s:url>" class="link">
            Salir</a>
12    </div>

14    <div id="center-column">
<!-- ############################################################ -->
16    <div class="top-bar">
          <a href="<s:url action="salirDelSistema"></s:url>" class="button
              ">Salir </a>
18        <h1>Fases</h1>
          <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
              Proyectos</a></div>
20    </div><br/>

22    <p>
        Las fases del proyecto son:
```

```
    </p>

    <s:form action="iteraciones">
      <s:select name="faseElegida" list="fases" listKey="id.nombre"
          listValue="id.nombre"></s:select>
      <s:submit value="Aceptar"></s:submit>
    </s:form>

<!-- ############################################################ -->
    </div>
    <div id="right-column">
      <strong class="h">INFO</strong>
      <div class="box"></div>
    </div>
</div>
```

jsp/planificar/fases.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>
  <div id="header">
    <a href="index.html" class="logo"><img src="img/logo.gif" width="101
        " height="29" alt="" /></a>
    <ul id="top-navigation">
      <li><span><span><a href="<s:url action="inicio"/>">Inicio</a></
          span></span></li>
      <li><span><span><a href="<s:url action="proyectosUsuario"/>">
          Proyecto</a></span></span></li>
      <li><span><span><a href="<s:url action="personalProyecto"/>">
          Personal</a></span></span></li>
      <li  class="active"><span><span><a href="<s:url action="
          iteraciones"/>">Planificar</a></span></span></li>
      <li><span><span><a href="actividadesRup">Actividades</a></span></
          span></li>
      <li><span><span><a href="rolesSistema">Rol</a></span></span></li>
      <li><span><span><a href="artefactosSistema">Artefactos</a></span
          ></span></li>
      <li><span><span><a href="tareasUsuario">Tareas</a></span></span></
          li>
      <li><span><span><a href="verPermisos">Seguridad</a></span></span
          ></li>
    </ul>
  </div>
```

jsp/planificar/header.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>
<link type="text/css" href="/PrototipoWeb/jsp/css/jquery-ui-1.8.2.custom
    .css" rel="Stylesheet" />
<script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery-1.4.2.
    min.js"></script>
<script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery-ui
    -1.8.2.custom.min.js"></script>
```

```
<script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery.ui.
    datepicker-es.js"></script>
<script type="text/javascript">
<!--
var x = $(document);
x.ready(inicializarEventos);

function inicializarEventos(){
  var x = $("#fechaInicio");
  x.datepicker();
}
//-->
</script>



<div id="middle">
    <div id="left-column">
      <h3>Menu</h3>
      <ul class="nav">
        <li><a href="<s:url action="nuevaIteracion"/>">Nueva Iteracion</
            a></li>
        <li><a href="<s:url action="proyectosUsuario"/>">Cambiar de
            Proyecto Actual</a></li>
      </ul>
    </div>

    <div id="center-column">
<!-- ##################################################################### -->
    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Nueva Iteracion</h1>
        <div class="breadcrumbs"><a href="#">Proyecto</a> / <a href="#">
            Proyectos</a></div>
    </div><br/>
    <p>
      Proporcione los datos de la nueva iteracion.
    </p>

    <s:form action="agregarIteracion">
      <s:textfield name="iteracion.nombre" label="Nombre"></s:textfield>
      <s:textarea name="iteracion.descripcion" label="Descripcion" cols=
          "30" rows="8"></s:textarea>
      <s:select name="faseElegida" list="fases" listKey="idFase"
          listValue="nombre"></s:select>
      <s:textfield id="fechaInicio" name="fechaInicio" label="Fecha
          Inicio"></s:textfield>
      <s:submit value="Aceptar"></s:submit>
    </s:form>

<!-- ##################################################################### -->
```

```
    </div>
    <div id="right-column">
      <strong class="h">INFO</strong>
      <div class="box"></div>
    </div>
</div>
```

jsp/planificar/nuevaIteracion.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>

<div id="middle">
    <div id="left-column">
      <h3>Menu</h3>
      <ul class="nav">
        <li><a href="<s:url action="nuevaIteracion"/>">Nueva Iteracion</
            a></li>
        <li><a href="<s:url action="datosProyectos"/>">Eliminar</a></li>
        <li><a href="<s:url action="listarProyectos"/>">Cambiar Proyecto
             Actual</a></li>
      </ul>
    </div>

    <div id="center-column">
<!-- ############################################################ -->
    <div class="top-bar">
        <a href="<s:url action="salirDelSistema"></s:url>" class="button
            ">Salir </a>
        <h1>Detalles de la Tarea</h1>
        <div class="breadcrumbs"><a href="#">Planificar</a> / <a href="#
            ">Iteraciones</a> / <a href="#">Detalles Tarea</a> </div>
    </div><br/>
    <p>
      Proporcione los datos para la nueva tarea:
    </p>
    <s:form>
      <s:textfield name="tarea.nombreTarea" label="Nombre de la Tarea: "
          ></s:textfield>
      <s:textfield name="tarea.fechaInicio" label="Fecha de Inicio"></s:
          textfield>
      <s:textfield name="tarea.fechaFin" label="Fecha de Termino"></s:
          textfield>
      <s:textfield name="tarea.porcentaje" label="Completado"></s:
          textfield>

      <s:select label="Disciplina" list="disciplinas"
        listValue="nombre" listKey="idDisciplina"
        size="5"></s:select>
      <s:select label="Actividad" list="actividades" listKey="
          idActividad"
        listValue="nombre" size="5"></s:select>

      <s:select label="Tarea Padre" list="tareas"
```

```
36          listValue="nombreTarea" listKey="idTarea" headerKey="0"
            headerValue="----------Ninguna tarea------"
38          size="5"></s:select>

40

        <s:optiontransferselect
42          name="responsables"
                leftTitle="Personal Asignado"
44              rightTitle="Personal Disponible"
                list="usuariosAsignados"
46              listKey="idUsuario"
                listValue="nombreCompleto"
48              multiple="true"
                headerKey="headerKey"
50              headerValue="--------------------------------"
                size="12"
52              emptyOption="false"
                doubleList="usuariosProyecto"
54              doubleListKey="idUsuario"
                doubleListValue="nombreCompleto"
56              doubleName="noresponsables"
                doubleHeaderKey="doubleHeaderKey"
58              doubleMultiple="true"
                doubleSize="12"
60              allowUpDownOnLeft="false"
                allowUpDownOnRight="false"
62              allowAddAllToLeft="false"
                allowAddAllToRight="false"
64              allowSelectAll="false"
        />
66

68      </s:form>

70  <!-- ############################################################ -->
      </div>
72    <div id="right-column">
        <strong class="h">INFO</strong>
74      <div class="box"> </div>
      </div>
76  </div>
```

jsp/planificar/detallesTarea.jsp

```
  <%@ page language="java" contentType="text/html; charset=UTF-8"
2     pageEncoding="UTF-8" %>
  <%@ taglib prefix="s" uri="/struts-tags" %>
4 <%@ taglib uri="http://tiles.apache.org/tags-tiles" prefix="tiles" %>
  <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://
     www.w3.org/TR/html4/loose.dtd">
6 <html>
  <head>
8 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
```

```
<title><tiles:getAsString name="pageTitle"/></title>
    <link href="<s:url value="/jsp/css/admin2.css"/>" rel="stylesheet"
          type="text/css"/>
<script type="text/javascript" src="/PrototipoWeb/jsp/js/jquery-1.4.2.js
    "></script>
</head>
<body>
Proyecto: <s:property value="#session.nombreProyecto"/><br></br>
Usuario: <s:property value="#session.nombreUsuario"/><br></br>
<div id="main">
<s:property value="#session.nombreProyecto"/>
  <!-- header -->
  <tiles:insertAttribute name="header"/>
  <!-- middle -->
  <tiles:insertAttribute name="middle"/>

  <!-- footer -->
  <tiles:insertAttribute name="footer"/>
</div>

</body>
</html>
```

jsp/planificar/estructuraGeneral.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://
    www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Error</title>
</head>
<body>
Error !!!
</body>
</html>
```

jsp/error/error.jsp

```
@CHARSET "UTF-8";
/* main styles */
body {
  margin:0;
  padding:0;
  background:#BBD9EE;
    color:#000;
  font-family:tahoma,arial,sans-serif;
  font-size:11px;
  }

form {
```

```
13    margin:0;
      padding:0
15    }
img {border:none;}
17
/*a {color:#060606;text-decoration: none}*/
19 a {color:#0101df;text-decoration: none}

21 a:hover {text-decoration: underline}

23 input {vertical-align:middle}

25 .floatleft {float:left !important}

27 .floatright {float:right !important}

29 .clear {clear:both !important}

31 .bold {font-weight:bold !important}

33 .normal {font-weight:normal !important}

35 .block {display:block !important}

37 input.text,
select,
39 textarea {
      font-family:arial,sans-serif;
41    color:#333;
      font-size:12px;
43    vertical-align:middle;
      }
45
input.text {
47    padding:1px 0 0 4px;
      height:14px;
49    font-weight:normal;
      }
51
/* main container */
53 #main {
      width:1300px;
55    margin:0 auto;
      }
57
/* header */
59 #header {
      position:relative;
61    width:1300px;
      height:109px;
63    background:url(../img/bg-header2.gif) no-repeat left bottom;
      }
```

```
65
/* site logo */
67 a.logo {
   position:absolute;
69 top:5px;
   left:20px;
71 }

73 /* header tabs */
#top-navigation {
75 position:absolute;
   top:60px;
77 left:20px;
   margin:0;
79 padding:0;
   list-style:none;
81 }
#top-navigation li {
83 float:left;
   margin:0 3px 0 0;
85 height:34px;
   background:url(../img/tab.gif) repeat-x top;
87 }
#top-navigation li a {
89 float:left;
   display:block;
91 height:20px;
   line-height:19px;
93 color:#606060;
   padding:4px 0 0 0;
95 }
#top-navigation li span {
97 float:left;
   background:url(../img/tab-left.gif) no-repeat left top;
99 }
#top-navigation li span span {
101 background:url(../img/tab-right.gif) no-repeat right top;
   padding:7px 10px 0 10px;
103 }
#top-navigation li.active {
105 padding:0;
   height:34px;
107 background:url(../img/tab-active.gif) repeat-x top;
   margin-right:2px;
109 }
#top-navigation li.active span {
111 background:url(../img/tab-active-left.gif) no-repeat left top;
   height:34px;
113 }
#top-navigation li.active span span {
115 background:url(../img/tab-active-right.gif) no-repeat right top;
   height:23px;
```

```
117    padding:11px 10px 0 10px;
       line-height:19px;
119    color:#606060;
       }
121
   /* middle */
123 #middle {
       float:left;
125    width:1200px;
       background:url(../img/bg-middle2.gif) repeat-y left;
127    /*background:url(../img/bg-middle.gif) repeat-x top;*/
       padding:0 13px 0 12px;
129    }

131 /* left column */
   #left-column {
133    float:left;
       padding:1px 14px 0 12px;
135    width:151px;
       }
137 /* right column */
   #right-column {
139    float:right;
       padding:0 9px 0 0;
141    width:133px;
       }
143 /* center column */
   #center-column {
145    float:left;
       width:800px;/*614*/
147    background:url(../img/bg-center-column2.jpg) no-repeat left top;
       min-height:700px;
149    padding:12px 16px 0 13px;
       }
151
   #center-column2 {
153    float:left;
       width:800px;/*614*/
155    min-height:700px;
       padding:12px 16px 0 13px;
157    }
   * html #center-column {height:584px;}
159
   /* footer */
161 #footer {
       float:left;
163    width:100%;
       background:url(../img/bg-footer2.gif) no-repeat;
165    height:15px;
       }
167
   /* left column styles */
```

```
169 #left-column a {color:#3E3E3E;}

171 #left-column h3 {
    font-size:11px;
173 margin:0;
    color:#fff;
175 background:url(../img/bg-left-header.gif) no-repeat left top;
    height:25px;
177 line-height:23px;
    padding:0 0 0 9px;
179 }

181 ul.nav {
    margin:0 0 11px 0;
183 border-bottom:2px solid #FF9600;
    background:#ECEFE7;
185 list-style:none;
    padding:0 2px;
187 }

189 ul.nav li {
    padding:4px 4px 6px 5px;
191 background:url(../img/bg-dotted.gif) repeat-x bottom;
    }
193
    ul.nav a {
195 padding:0 0 0 12px;
    background:url(../img/arrow.gif) no-repeat 0 4px;
197 }

199 ul.nav a:hover {
    font-weight:bold;
201 }

203 ul.nav li.last {
    background:#A9A9F5;
205 }

207 #left-column .link {
    display:block;
209 width:142px;
    height:25px;
211 background:url(../img/bg-left-link.gif);
    margin:0 0 4px 0;
213 font-weight:bold;
    padding:0 0 0 9px;
215 line-height:25px;
    color:#60635A;
217 }

219 /* center column styles */
    .top-bar {
```

```
221   float:left;
      width:750px;
223   /*border-left:2px solid #f70;*/
      border-bottom:2px solid #f70; /* esto fue agregado por paul*/
225   padding:0 0 0 9px;
      margin:0 10px 20px 0;
227   }

229 /* text page header */
    .top-bar h1 {
231   font:20px/21px verdana,sans-serif;
      color:#43729F;
233   margin:0 0 4px 0;
      }
235
    /* orange button */
237 .top-bar a.button {
      float:right;
239   display:block;
      width:75px;
241   height:35px;
      text-align:center;
243   color:#fff;
      text-transform:uppercase;
245   font-weight:bold;
      line-height:27px;
247   background:url(../img/bg-orange-button.gif) no-repeat;
      }
249
    /* bar with select */
251 .select-bar {
      clear:both;
253   border-top:2px solid #f70;
      border-bottom:2px solid #f70;
255   padding:5px 0 3px 0;
      margin:0 0 17px 0;
257   }

259 .select-bar select {width:145px;margin:0 2px;}

261 /* table container */
    div.table {
263   float:left;
      position:relative;
265   width:890px;
      margin:0 0 37px 0;
267   }

269 table.listing {
      border-bottom:1px solid #9097A9;
271   width:890px;
      padding:0;
```

```
273   margin:0;
      border:1px solid #9097A9;
275   }

277 table.listing th {
      border-top:0 !important;
279   }

281 table.listing th.full {
      border-left:0;
283   border-right:0 !important;
      text-align:left;
285   text-transform:uppercase;
    }
287
    div.table img.left {
289   /*position:absolute;*/
      position:relative;
291   top:0;
      left:0;
293   }

295 div.table img.right {
        /*position:absolute;*/
297   position:relative;
      top:0;
299   right:1px;
      }
301
    /* table styles */
303 table.listing td,
    table.listing th {
305   border:1px solid #fff;
      text-align:center;
307   }

309 table.listing th {
      background:#9097A9;
311   color:#fff;
      padding:5px;
313   }

315 table.listing td {
      background:#D8D8D8;
317   color:#000;
      padding:3px 5px;
319   }
    table.listing .bg td {
321   background:#ECECEC;
      }
323
    table.listing .white td {
```

```
325   background:#fff;
      }
327
table.listing .first {border-left:0px solid #9097A9;text-align:left;}
329 table.listing .last {border-right:0px solid #9097A9;}

331 table.listing th.first {background:#9097A9 url(../img/bg-th-left.gif) no
      -repeat left top;border-left:0;}
    table.listing th.last {background:#9097A9 url(../img/bg-th-right.gif) no
      -repeat right top;border-right:0;}
333
    table.listing .style1 {font-weight:bold;color:#FF7A00;}
335 table.listing .style2 {font-weight:bold;padding-left:16px;}
    table.listing .style3 {padding-left:25px;}
337 table.listing .style4 {padding-left:35px;}
    table.form .last {padding:1px 0 1px 5px;text-align:left;}
339 table.form th,
    table.form td {padding-left:10px;}
341 table.form input.text {width:262px}

343 /* table select */
    div.table .select {
345   float:right;
      margin:2px 1px 0 0;
347   width:176px;
      height:25px;
349   background:#9097A9 url(../img/bg-select.gif);
      color:#fff;
351   }
    div.table .select strong {
353   float:left;
      padding:5px 0 0 5px;
355   }
    div.table .select select {
357   float:right;
      width:78px;
359   margin:2px 3px 0 0;
      text-align:right;
361   }

363 /* right column header */
    #right-column .h {
365   float:left;
      background:#7E878A;
367   border:1px solid #B8B8B8;
      border-bottom:0;
369   padding:3px 10px;
      color:#fff;
371   text-transform:uppercase;
      }
373 /* right column box */
    #right-column .box {
```

```
375   float:left;
      width:121px;
377   padding:5px;
      border:1px solid #B8B8B8;
379   background:#EBEBEB;
      margin:0 0 15px 0;
381   }

383 /* right column buttons */
    .buttons {
385   clear:both;
      text-align:center;
387   padding:30px 0 15px 0;
      }
389 .buttons input {margin:0 0 6px 0;}


391
    div img {
393   margin-top: .5em;
      margin-bottom: .5em;
395   margin-left: 1em;
      margin-right: .5em;
397   padding: 10% 40%
    }
399
    /*
401 div label {
      width: 25%;
403   float: left;
    }
405 form input{
      background:url(../img/bg-orange-button.gif) no-repeat;
407 }*/
```

jsp/css/admin2.css

Apéndice C

Script SQL

```
CREATE TABLE `prototipov6`.`Accion` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `nombre` varchar(50) NOT NULL,
  `descripcion` varchar(250) DEFAULT NULL,
  `categoria` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `nombreAccion` (`nombre`)
) ENGINE=InnoDB AUTO_INCREMENT=156 DEFAULT CHARSET=latin1;

CREATE TABLE `prototipov6`.`Actividad` (
  `idActividad` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `Rol_nombreRol` varchar(45) NOT NULL,
  `Disciplina_idDisciplina` int(10) unsigned NOT NULL,
  `nombre` varchar(255) DEFAULT NULL,
  `descripcion` text,
  `urlDescripcion` varchar(255) DEFAULT NULL,
  `informacionGeneral` text NOT NULL,
  PRIMARY KEY (`idActividad`),
  KEY `Actividad_FKIndex1` (`Disciplina_idDisciplina`),
  KEY `Actividad_FKIndex2` (`Rol_nombreRol`),
  CONSTRAINT `Actividad_ibfk_1` FOREIGN KEY (`Disciplina_idDisciplina`)
      REFERENCES `Disciplina` (`idDisciplina`) ON UPDATE CASCADE,
  CONSTRAINT `Actividad_ibfk_2` FOREIGN KEY (`Rol_nombreRol`) REFERENCES
      `Rol` (`nombreRol`) ON UPDATE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=14 DEFAULT CHARSET=latin1;

CREATE TABLE `prototipov6`.`Artefacto` (
  `nombre` varchar(45) NOT NULL,
  `descripcion` varchar(1000) DEFAULT NULL,
  `urlDescripcion` varchar(255) DEFAULT NULL,
```

```
29    `Rol_nombreRol` varchar(45) NOT NULL,
      `proposito` text,
31    `timing` text,
      `tailoring` text,
33    `representacionUML` varchar(500) NOT NULL,
      PRIMARY KEY (`nombre`),
35    KEY `fk_rol` (`Rol_nombreRol`),
      CONSTRAINT `fk_rol` FOREIGN KEY (`Rol_nombreRol`) REFERENCES `Rol` (`
          nombreRol`) ON UPDATE CASCADE
37  ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

39  CREATE TABLE `prototipov6`.`ArtefactoContieneArtefacto` (
      `Artefacto_nombre` varchar(45) NOT NULL,
41    PRIMARY KEY (`Artefacto_nombre`),
      KEY `Artefacto_has_Artefacto_FKIndex1` (`Artefacto_nombre`),
43    KEY `Artefacto_has_Artefacto_FKIndex2` (`Artefacto_nombre`)
    ) ENGINE=MyISAM DEFAULT CHARSET=latin1;
45
    CREATE TABLE `prototipov6`.`ArtefactoProducidoActividad` (
47    `Actividad_idActividad` int(10) unsigned NOT NULL,
      `Artefacto_nombre` varchar(45) NOT NULL,
49    PRIMARY KEY (`Actividad_idActividad`,`Artefacto_nombre`),
      KEY `Artefacto_has_Actividad_FKIndex1` (`Artefacto_nombre`),
51    KEY `Artefacto_has_Actividad_FKIndex2` (`Actividad_idActividad`),
      CONSTRAINT `ArtefactoProducidoActividad_ibfk_1` FOREIGN KEY (`
          Artefacto_nombre`) REFERENCES `Artefacto` (`nombre`) ON UPDATE
          CASCADE,
53    CONSTRAINT `ArtefactoProducidoActividad_ibfk_2` FOREIGN KEY (`
          Actividad_idActividad`) REFERENCES `Actividad` (`idActividad`) ON
          UPDATE CASCADE
    ) ENGINE=InnoDB DEFAULT CHARSET=latin1;
55
    CREATE TABLE `prototipov6`.`ArtefactoRequeridoActividad` (
57    `Actividad_idActividad` int(10) unsigned NOT NULL,
      `Artefacto_nombre` varchar(45) NOT NULL,
59    PRIMARY KEY (`Actividad_idActividad`,`Artefacto_nombre`),
      KEY `Artefacto_has_Actividad_FKIndex1` (`Artefacto_nombre`),
61    KEY `Artefacto_has_Actividad_FKIndex2` (`Actividad_idActividad`),
      CONSTRAINT `ArtefactoRequeridoActividad_ibfk_1` FOREIGN KEY (`
          Artefacto_nombre`) REFERENCES `Artefacto` (`nombre`) ON UPDATE
          CASCADE,
63    CONSTRAINT `ArtefactoRequeridoActividad_ibfk_2` FOREIGN KEY (`
          Actividad_idActividad`) REFERENCES `Actividad` (`idActividad`) ON
          UPDATE CASCADE
    ) ENGINE=InnoDB DEFAULT CHARSET=latin1;
65
    CREATE TABLE `prototipov6`.`Disciplina` (
67    `idDisciplina` int(10) unsigned NOT NULL AUTO_INCREMENT,
      `nombre` varchar(255) DEFAULT NULL,
69    `descripcion` varchar(255) DEFAULT NULL,
      PRIMARY KEY (`idDisciplina`)
71  ) ENGINE=InnoDB AUTO_INCREMENT=8 DEFAULT CHARSET=latin1;
```

```
73 CREATE TABLE `prototipov6`.`Ejemplar` (
     `Artefacto_nombre` varchar(45) NOT NULL,
75   `idEjemplar` int(10) unsigned NOT NULL,
     `Proyecto_idProyecto` int(10) unsigned NOT NULL,
77   `urlUbicacion` int(10) unsigned DEFAULT NULL,
     `fechaCreacion` date DEFAULT NULL,
79   PRIMARY KEY (`Artefacto_nombre`,`idEjemplar`,`Proyecto_idProyecto`),
     KEY `Ejemplar_FKIndex1` (`Artefacto_nombre`),
81   KEY `Ejemplar_FKIndex2` (`Proyecto_idProyecto`),
     CONSTRAINT `Ejemplar_ibfk_1` FOREIGN KEY (`Artefacto_nombre`)
        REFERENCES `Artefacto` (`nombre`) ON DELETE NO ACTION ON UPDATE NO
         ACTION,
83   CONSTRAINT `Ejemplar_ibfk_2` FOREIGN KEY (`Proyecto_idProyecto`)
        REFERENCES `Proyecto` (`idProyecto`) ON DELETE NO ACTION ON UPDATE
         NO ACTION
   ) ENGINE=InnoDB DEFAULT CHARSET=latin1;
85
   CREATE TABLE `prototipov6`.`EjemplarArtefacto` (
87   `Proyecto_idProyecto` int(10) unsigned NOT NULL,
     `Artefacto_nombre` varchar(50) NOT NULL,
89   `nombreArchivo` varchar(50) NOT NULL,
     `tipoContenido` varchar(50) NOT NULL,
91   `contenidoArchivo` mediumblob NOT NULL,
     `fechaCreacion` date NOT NULL,
93   PRIMARY KEY (`Proyecto_idProyecto`,`Artefacto_nombre`),
     KEY `fk_artefacto` (`Artefacto_nombre`),
95   CONSTRAINT `fk_artefacto` FOREIGN KEY (`Artefacto_nombre`) REFERENCES
        `Artefacto` (`nombre`) ON UPDATE CASCADE,
     CONSTRAINT `fk_proyecto` FOREIGN KEY (`Proyecto_idProyecto`)
        REFERENCES `Proyecto` (`idProyecto`) ON UPDATE CASCADE
97 ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

99 CREATE TABLE `prototipov6`.`Fase` (
     `idFase` int(10) unsigned NOT NULL AUTO_INCREMENT,
101  `nombre` varchar(25) NOT NULL,
     `descripcion` varchar(255) DEFAULT NULL,
103  PRIMARY KEY (`idFase`),
     UNIQUE KEY `NOMBREFASE` (`nombre`)
105 ) ENGINE=InnoDB AUTO_INCREMENT=5 DEFAULT CHARSET=latin1;

107 CREATE TABLE `prototipov6`.`Iteracion` (
     `numIteracion` int(10) unsigned NOT NULL,
109  `Proyecto_idProyecto` int(10) unsigned NOT NULL,
     `Fase_idFase` int(10) unsigned NOT NULL,
111  `descripcion` varchar(255) DEFAULT NULL,
     `fechaInicio` date DEFAULT NULL,
113  `indice` int(10) unsigned DEFAULT NULL,
     `nombre` varchar(25) DEFAULT NULL,
115  PRIMARY KEY (`numIteracion`,`Proyecto_idProyecto`),
     KEY `Iteracion_FKIndex1` (`Proyecto_idProyecto`),
117  KEY `Iteracion_FKIndex2` (`Fase_idFase`),
```

```
  CONSTRAINT `fk_fase` FOREIGN KEY (`Fase_idFase`) REFERENCES `Fase` (`
      idFase`),
  CONSTRAINT `Iteracion_ibfk_1` FOREIGN KEY (`Proyecto_idProyecto`)
      REFERENCES `Proyecto` (`idProyecto`) ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `prototipov6`.`PadreTareaHijo` (
  `numTarea` int(10) unsigned NOT NULL,
  `numIteracion` int(10) unsigned NOT NULL,
  `numProyecto` int(10) unsigned NOT NULL,
  `numTareaHijo` int(10) unsigned NOT NULL,
  `numIteracionH` int(10) unsigned NOT NULL,
  `numProyectoH` int(10) unsigned NOT NULL,
  PRIMARY KEY (`numTarea`,`numIteracion`,`numProyecto`,`numTareaHijo`,`
      numIteracionH`,`numProyectoH`),
  KEY `FKE849CD61F2980ADC` (`numTarea`,`numIteracion`,`numProyecto`),
  KEY `FKE849CD615D6FE1EC` (`numTareaHijo`,`numIteracionH`,`numProyectoH
      `),
  KEY `padre` (`numProyecto`,`numIteracion`),
  KEY `fk_padre` (`numTarea`,`numProyecto`,`numIteracion`),
  KEY `fk_hijo` (`numTareaHijo`,`numProyectoH`,`numIteracionH`),
  CONSTRAINT `fk_hijo` FOREIGN KEY (`numTareaHijo`, `numProyectoH`, `
      numIteracionH`) REFERENCES `Tarea` (`numTarea`, `
      Iteracion_Proyecto_idProyecto`, `Iteracion_numIteracion`) ON
      DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `fk_padre` FOREIGN KEY (`numTarea`, `numProyecto`, `
      numIteracion`) REFERENCES `Tarea` (`numTarea`, `
      Iteracion_Proyecto_idProyecto`, `Iteracion_numIteracion`) ON
      DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1 ROW_FORMAT=FIXED;

CREATE TABLE `prototipov6`.`ParticipanteProyecto` (
  `Usuario_CURP` varchar(20) NOT NULL,
  `Proyecto_idProyecto` int(10) unsigned NOT NULL,
  PRIMARY KEY (`Usuario_CURP`,`Proyecto_idProyecto`),
  KEY `Usuario_has_Proyecto_FKIndex1` (`Usuario_CURP`),
  KEY `Usuario_has_Proyecto_FKIndex2` (`Proyecto_idProyecto`),
  CONSTRAINT `ParticipanteProyecto_ibfk_1` FOREIGN KEY (`Usuario_CURP`)
      REFERENCES `Usuario` (`CURP`) ON UPDATE CASCADE,
  CONSTRAINT `ParticipanteProyecto_ibfk_2` FOREIGN KEY (`
      Proyecto_idProyecto`) REFERENCES `Proyecto` (`idProyecto`) ON
      DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `prototipov6`.`Permiso` (
  `idAccion` int(11) NOT NULL,
  `rol` varchar(45) NOT NULL,
  `permitido` tinyint(1) NOT NULL,
  PRIMARY KEY (`idAccion`,`rol`) USING BTREE,
  KEY `fk_roles` (`rol`),
  CONSTRAINT `fk_accion` FOREIGN KEY (`idAccion`) REFERENCES `Accion` (`
      id`) ON DELETE CASCADE ON UPDATE CASCADE,
```

```
  CONSTRAINT `fk_roles` FOREIGN KEY (`rol`) REFERENCES `Rol` (`nombreRol
      `)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `prototipov6`.`Plantilla` (
  `Artefacto_nombre` varchar(45) NOT NULL,
  `Proyecto_idProyecto` int(10) unsigned NOT NULL,
  `nombreArchivo` varchar(255) DEFAULT NULL,
  `tipoArchivo` varchar(255) DEFAULT NULL,
  `contenido` mediumblob,
  `fechaCreacion` date DEFAULT NULL,
  PRIMARY KEY (`Artefacto_nombre`,`Proyecto_idProyecto`),
  KEY `Plantilla_FKIndex1` (`Artefacto_nombre`),
  KEY `Plantilla_FKIndex2` (`Proyecto_idProyecto`),
  CONSTRAINT `Plantilla_ibfk_1` FOREIGN KEY (`Artefacto_nombre`)
      REFERENCES `Artefacto` (`nombre`) ON UPDATE CASCADE,
  CONSTRAINT `Plantilla_ibfk_2` FOREIGN KEY (`Proyecto_idProyecto`)
      REFERENCES `Proyecto` (`idProyecto`) ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `prototipov6`.`Proyecto` (
  `idProyecto` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `Usuario_CURP` varchar(20) NOT NULL,
  `nombre` varchar(255) DEFAULT NULL,
  `fechaInicio` date DEFAULT NULL,
  `fechaFin` date DEFAULT NULL,
  `descripcion` varchar(255) DEFAULT NULL,
  `cliente` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`idProyecto`),
  KEY `Proyecto_FKIndex1` (`Usuario_CURP`),
  CONSTRAINT `Proyecto_ibfk_1` FOREIGN KEY (`Usuario_CURP`) REFERENCES `
      Usuario` (`CURP`) ON UPDATE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=latin1;

CREATE TABLE `prototipov6`.`Rol` (
  `nombreRol` varchar(45) NOT NULL,
  `descripcion` text,
  `urlDescripcion` varchar(255) DEFAULT NULL,
  `categoria` varchar(45) DEFAULT NULL,
  `habilidades` text NOT NULL,
  `criterioAsignacion` text NOT NULL,
  PRIMARY KEY (`nombreRol`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `prototipov6`.`Tarea` (
  `numTarea` int(10) unsigned NOT NULL,
  `Iteracion_Proyecto_idProyecto` int(10) unsigned NOT NULL,
  `Iteracion_numIteracion` int(10) unsigned NOT NULL,
  `Actividad_idActividad` int(10) unsigned NOT NULL,
  `nombreTarea` varchar(255) DEFAULT NULL,
  `fechaInicio` date DEFAULT NULL,
  `fechaFin` date DEFAULT NULL,
```

```
  `hito` int(10) unsigned DEFAULT NULL,
  `porcentaje` int(10) unsigned DEFAULT NULL,
  `pGroup` int(10) unsigned DEFAULT NULL,
  `pOpen` int(10) unsigned DEFAULT NULL,
  `pCaption` varchar(45) DEFAULT NULL,
  `indice` int(10) unsigned DEFAULT NULL,
  `Usuario_CURP` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`numTarea`,`Iteracion_Proyecto_idProyecto`,`
      Iteracion_numIteracion`),
  KEY `Tarea_FKIndex` (`Actividad_idActividad`),
  KEY `Tarea_FKIndex2` (`Iteracion_numIteracion`,`
      Iteracion_Proyecto_idProyecto`),
  KEY `Tarea_FKIndex3` (`Actividad_idActividad`),
  KEY `FK4CD86E1FC8B99D` (`Iteracion_numIteracion`,`
      Iteracion_Proyecto_idProyecto`),
  KEY `FK4CD86E15D0CA319` (`Usuario_CURP`),
  CONSTRAINT `FK4CD86E15D0CA319` FOREIGN KEY (`Usuario_CURP`) REFERENCES
       `Usuario` (`CURP`),
  CONSTRAINT `Tarea_ibfk_1` FOREIGN KEY (`Actividad_idActividad`)
     REFERENCES `Actividad` (`idActividad`),
  CONSTRAINT `Tarea_ibfk_2` FOREIGN KEY (`Iteracion_numIteracion`, `
      Iteracion_Proyecto_idProyecto`) REFERENCES `Iteracion` (`
      numIteracion`, `Proyecto_idProyecto`) ON DELETE CASCADE ON UPDATE
      CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `prototipov6`.`TareaPredecesor` (
  `numTarea` int(10) unsigned NOT NULL,
  `numIteracion` int(10) unsigned NOT NULL,
  `numProyecto` int(10) unsigned NOT NULL,
  `numIteracionP` int(10) unsigned NOT NULL,
  `numProyectoP` int(10) unsigned NOT NULL,
  `numTareaPredecesor` int(10) unsigned NOT NULL,
  PRIMARY KEY (`numTarea`,`numIteracion`,`numProyecto`,`numIteracionP`,`
      numProyectoP`,`numTareaPredecesor`),
  KEY `new_fk_constraint` (`numTareaPredecesor`,`numProyectoP`,`
      numIteracionP`),
  CONSTRAINT `fk_tarea` FOREIGN KEY (`numTarea`, `numIteracion`, `
      numProyecto`) REFERENCES `Tarea` (`numTarea`, `
      Iteracion_Proyecto_idProyecto`, `Iteracion_numIteracion`),
  CONSTRAINT `new_fk_constraint` FOREIGN KEY (`numTareaPredecesor`, `
      numProyectoP`, `numIteracionP`) REFERENCES `Tarea` (`numTarea`, `
      Iteracion_Proyecto_idProyecto`, `Iteracion_numIteracion`) ON
      DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `prototipov6`.`TareaUsuario` (
  `Usuario_CURP` varchar(20) NOT NULL,
  `Tarea_numTarea` int(10) unsigned NOT NULL,
  `Tarea_Iteracion_numIteracion` int(10) unsigned NOT NULL,
  `Tarea_Iteracion_Proyecto_idProyecto` int(10) unsigned NOT NULL,
  PRIMARY KEY (`Usuario_CURP`,`Tarea_numTarea`,`
```

```
    Tarea_Iteracion_numIteracion`,`Tarea_Iteracion_Proyecto_idProyecto
    `),
  KEY `Tarea_has_Usuario_FKIndex1` (`Tarea_numTarea`,`
    Tarea_Iteracion_Proyecto_idProyecto`,`Tarea_Iteracion_numIteracion
    `),
  KEY `Tarea_has_Usuario_FKIndex2` (`Usuario_CURP`),
  CONSTRAINT `TareaUsuario_ibfk_1` FOREIGN KEY (`Tarea_numTarea`, `
    Tarea_Iteracion_Proyecto_idProyecto`, `
    Tarea_Iteracion_numIteracion`) REFERENCES `Tarea` (`numTarea`, `
    Iteracion_Proyecto_idProyecto`, `Iteracion_numIteracion`) ON
    DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `TareaUsuario_ibfk_2` FOREIGN KEY (`Usuario_CURP`)
    REFERENCES `Usuario` (`CURP`) ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `prototipov6`.`Usuario` (
  `CURP` varchar(20) NOT NULL,
  `nombre` varchar(20) DEFAULT NULL,
  `ape_paterno` varchar(45) DEFAULT NULL,
  `ape_materno` varchar(45) DEFAULT NULL,
  `telefono` varchar(20) DEFAULT NULL,
  `correo` varchar(45) DEFAULT NULL,
  `login` varchar(45) DEFAULT NULL,
  `pass_word` varchar(45) DEFAULT NULL,
  PRIMARY KEY (`CURP`),
  UNIQUE KEY `correo` (`correo`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `prototipov6`.`UsuarioRol` (
  `Usuario_CURP` varchar(20) NOT NULL,
  `Rol_nombreRol` varchar(45) NOT NULL,
  PRIMARY KEY (`Usuario_CURP`,`Rol_nombreRol`),
  KEY `Usuario_has_Rol_FKIndex1` (`Usuario_CURP`),
  KEY `Usuario_has_Rol_FKIndex2` (`Rol_nombreRol`),
  CONSTRAINT `UsuarioRol_ibfk_1` FOREIGN KEY (`Usuario_CURP`) REFERENCES
      `Usuario` (`CURP`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `UsuarioRol_ibfk_2` FOREIGN KEY (`Rol_nombreRol`)
    REFERENCES `Rol` (`nombreRol`) ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

scriptBD.txt

Bibliografía

[1] Paulino Ramirez. Propuesta de proyecto terminal. *Prototipo Web para Gestionar Proyectos de Software*, 2010.

[2] Ivar Jacobson, Grady Booch, and James Rumbaugh. *El Proceso Unificado de Desarrollo de Software*. Addison Wesley, 2000.

[3] Richard Helm, Ralph Johnson, and John Vlissides. *Patrones de Diseño*. PEARSON Addison Wesley, 2008.

[4] Adam Drozdek. *Estructura de datos y algoritmos en Java*. THOMSON, 2007.

[5] Robert C. Martin. *UML para programadores Java*. PEARSON Prentice Hall, 2009.

[6] Antonio J. Martin Sierra. *Programador Certificado Java 2*. Alfa Omega Ra-Ma, 2 edition, 2008.

[7] Ian Gilfillan. *La Biblia de MySQL*. ANAYA, 2003.

[8] Ian Roughley. *Practical Apache Struts 2 Web 2.0 Projects*. Apress, 2007.

[9] Budi Kurniawan. *Struts 2 Design and Programming: A Tutorial*. BrainySoftware, 2008.

[10] Abraham Silberschatz, Henry F. Korth, and S. Sudarshan. *Fundamentos de Bases de Datos*. McGraw Hill, 4 edition, 2002.

[11] Gavin King, Christian Bauer, Max Rydahl Andersen, Emmanuel Bernard, and Steve Ebersole. Hibernate reference documentation, May.

[12] Emmanuel Bernard. Hibernate annotations reference guide.

[13] Emmanuel Bernard, Steve Ebersole, and Gavin King. Hibernate entity manager 1 user guide.

[14] Bear Bibeault and Yehuda Katz. *jQuery in Action*. MANNING, 2008.

[15] Ian Roughley. Starting struts 2.

[16] Donald Brow and Chad Michael Davis. *Struts 2 in Action*. MANNING, 2007.

Universidad Autónoma Metropolitana
Unidad Azcapotzalco

División de Ciencias Básicas e Ingeniería

Manual del Usuario

México, D.F. **Enero de 2011**

Índice general

Índice de figuras

Índice de tablas

Introducción

0.1. Acceso a la aplicación e inicio de sesión

Para acceder a la aplicación es necesario poner el url:

```
http://IP:puerto/PrototipoWeb/
```

Donde el IP corresponde a la dirección del servidor donde está la aplicación y puerto es el número de puerto en la cual se atiende la petición. Generalmente es el puerto 8080 u 80. Tras poner dicha url, se presentará una pantalla de bienvenida, tal como se ve en la figura 1.

Figura 1: Pantalla de bienvenida al sistema.

Para poder iniciar sesión hacer clic en el submenú que dice *Entrar*, tras lo cual se mostrará la pantalla de la figura 2. En esta pantalla hay que introducir los datos requeridos, el login, que es el correo electrónico proporcionado al momento de darse de alta y el password. Una vez introducidos los datos hay que hacer clic en el el botón *Validar*.

Figura 2: Pantalla de inicio se sesión.

0.2. Administración de Proyectos

Tras validarse en el sistema se presentará la pantalla de la figura 3. En esta pantalla tiene que elegir uno de los proyectos, esto es importante, ya que todas las otras opciones de los menús dependen de que se haya elegido un proyecto.

Figura 3: Pantalla para elegir proyecto.

En la pantalla 3 hay una descripción resumida de los proyectos, los cuales están en forma de lista. Debajo de cada proyecto hay dos palabras resaltadas de azul. La que dice *ver Detalles*, nos permite ver información detallada del proyecto sin poder modificarla, la pantalla que se muestra después de hacer clic sobre esta palabra se muestra en la figura 4. Desde esta pantalla podemos regresar a la anterior haciendo clic en *Regresar*.

También desde la pantalla 3, hay otra palabra resaltada en azul que dice *Modificar/Eliminar*, al hacer clic sobre estas palabras se presenta la pantalla de la figura 5. Desde esta pantalla podemos modificar la información del proyecto y después guardar los cambios o en su defecto eliminar el proyecto del sistema. Si no desea realizar ninguna acción basta con hacer clic en *Regresar*, y regresará a la pantalla 3.

Por otra parte, para dar de alta un nuevo proyecto desde la pantalla 3, en el submenú del lado izquierdo tiene la opción *Nuevo*, al hacer clic sobre esta opción se mostrará la pantalla de la figura 6.

Figura 4: Detalles del proyecto.

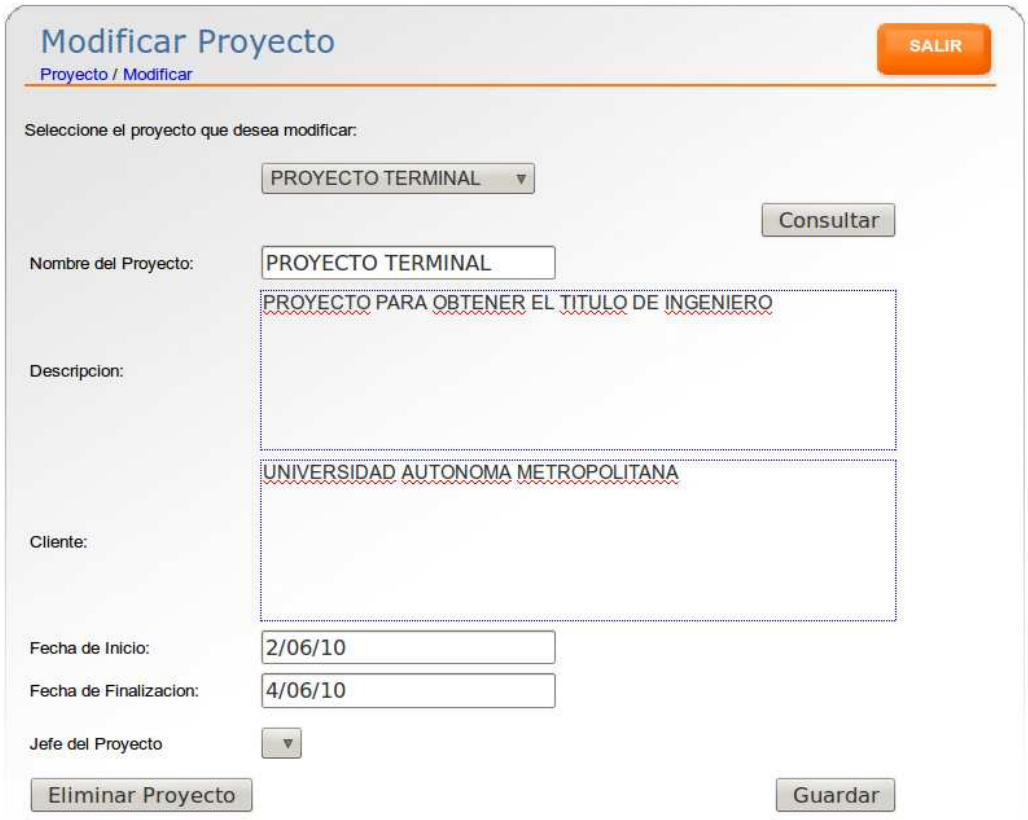

Figura 5: Pantalla de modificación o eliminación de proyecto.

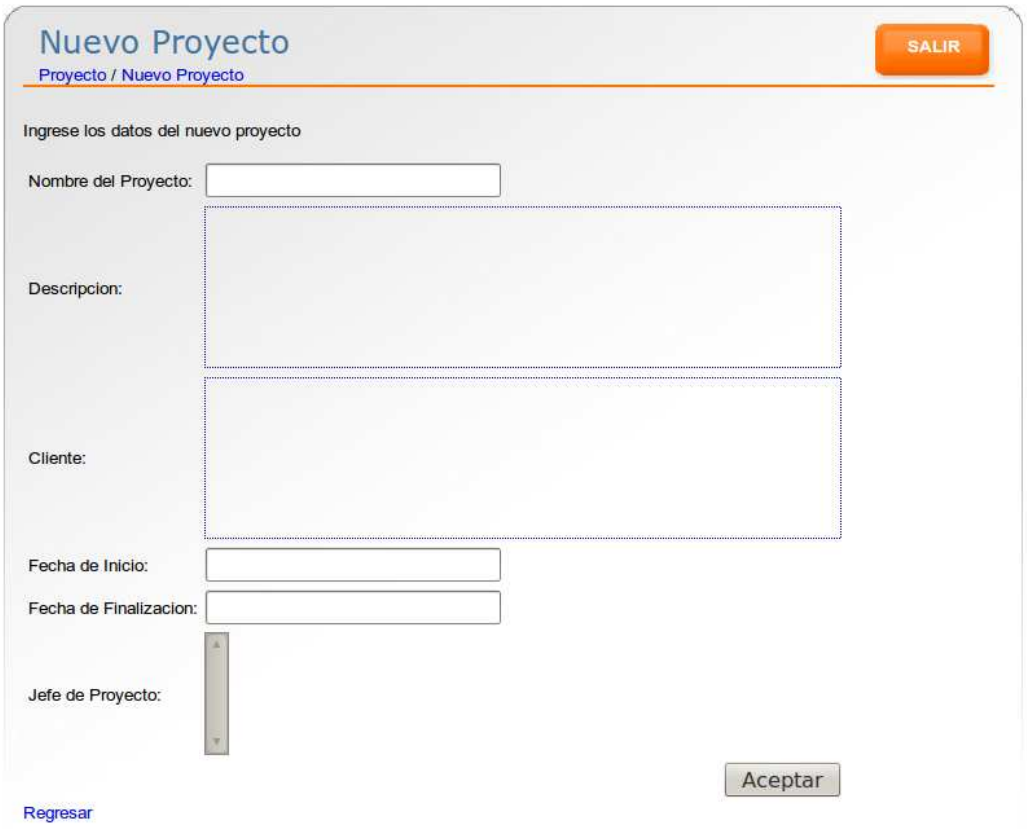

Figura 6: Pantalla para dar de alta un nuevo proyecto.

0.3. Administración del Personal

Después de haber elegido un proyecto podrá hacer uso de este menú. Al hacer clic sobre el menú *Personal* se mostrará la pantalla de la figura 7. Es un listado de los usuarios que están participando en el proyecto seleccionado en la pantalla 3. Debajo del nombre de cada usuario está la palabra *Detalles* resaltada de azul, al hacer clic sobre el mismo se presenta la pantalla de la figura 8, con información del usuario.

Figura 7: Lista de personal del proyecto.

Figura 8: Detalles del usuario elegido.

Por otra parte del lado izquierdo de la pantalla hay un conjunto de opciones que son: Pesonal del Proyecto, Agregar/Eliminar Personal, Eliminar/Modificar y Nuevo Personal. Las cuales explicaremos a continuación.

0.3.1. Agregar/Eliminar Personal

Esta opción muestra una pantalla desde la cual se agrega o elimina personal del proyecto actual. La pantalla se muestra en la figura 9. En esta pantalla de lado izquier-

do se tiene un listado de los usuarios que no están asignados actualmente al proyecto y de lado derecho el listado del personal actual del proyecto. Para agregar un personal al proyecto actual basta con seleccionarlo y hacer clic en la flecha iquierda, para quitar uno hay que seleccionarlo y hacer clic sobre la flecha derecha. Posteriormente hay que guardar los cambios seleccionando todos los elementos de la lista del personal del proyecto.

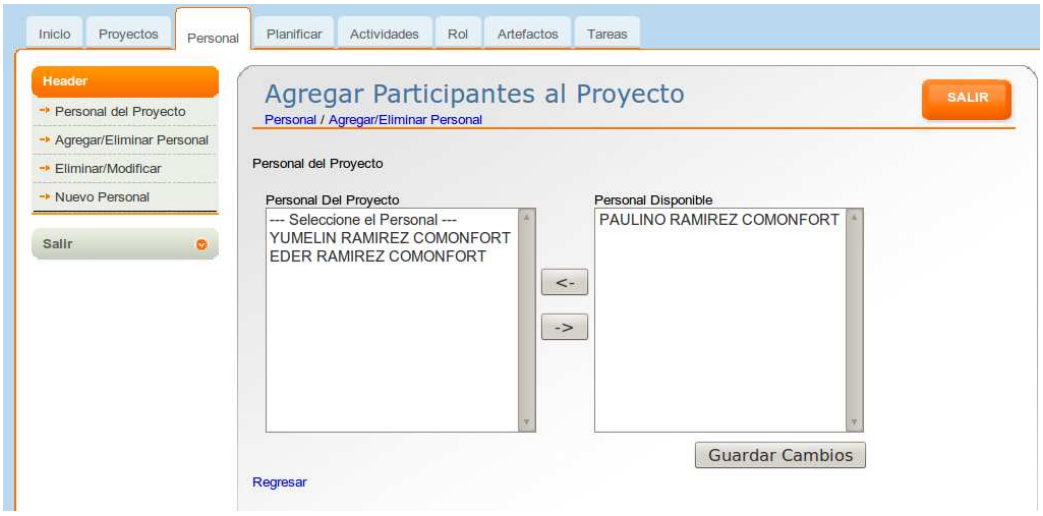

Figura 9: Pantalla para agregar o eliminar personal del proyecto actual.

0.3.2. Modificar/Eliminar

Con esta opción se pueden modificar los datos de un usuario del sistema, la pantalla se ve en la figura 10. Para ello primeramente se tiene que elegir un usuario del combo que se presenta y después hacer clic en el botón *Consultar*, tras lo cual se pueden modificar sus datos. Desde esta pantalla se hace la asignación de roles al usuario, se le pueden quitar o poner roles. Para asignarle un rol basta con hacer clic en rol deseado en la lista *Roles Disponibles* y después hacer clic sobre la flecha izquierda, para quitarle un rol es el proceso inverso. Para guardar los cambios hay que seleccionar todos los roles en la lista Rol Asignado y después hacer clic en *Guardar cambios*. Despueś de guardar los cambios se mostrará la pantalla 11 con los datos del usuario actualizado. En esta pantalla también se puede eliminar a un usuario con el botón *Eliminar*.

Por último con la opción *Nuevo Personal* se muestra la pantalla de la figura 12, desde la cual se puede dar de alta un nuevo usuario proporcionando los datos requeridos.

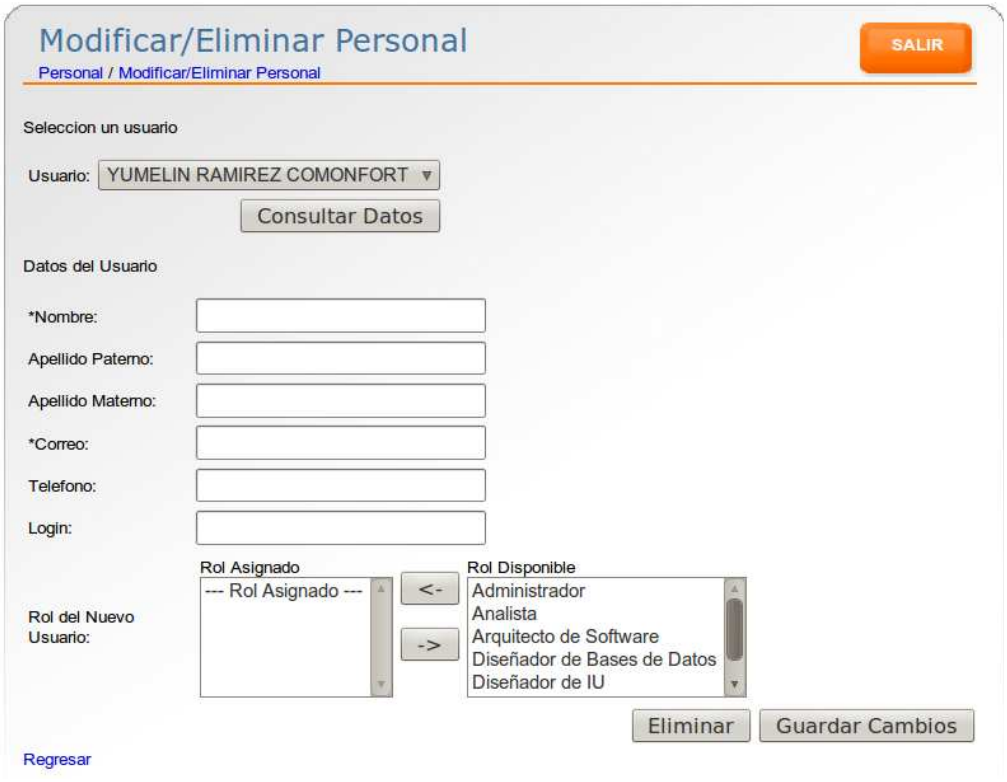

Figura 10: Pantalla para modificación o eliminación de usuario del proyecto.

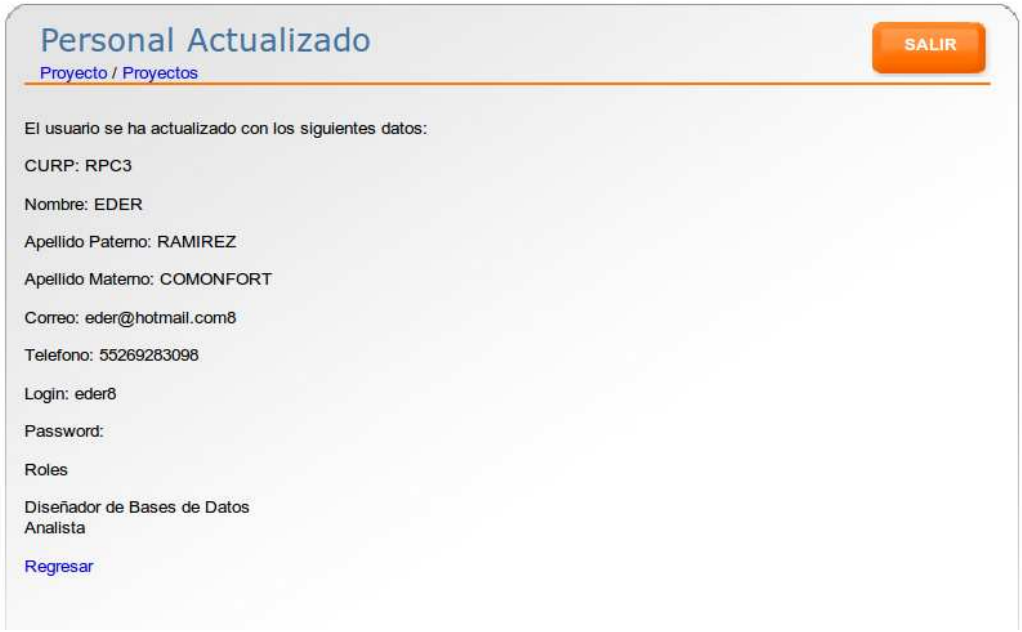

Figura 11: Datos actualizados del personal.

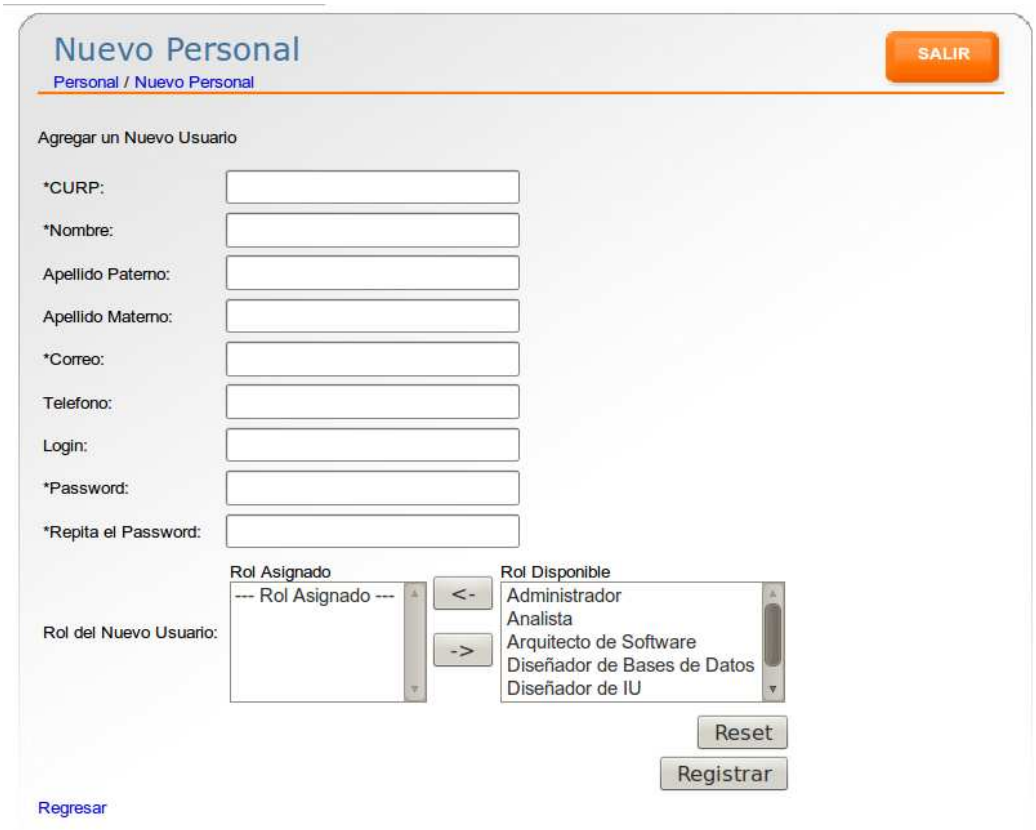

Figura 12: Registro de un nuevo personal.

0.4. Planificación del Proyecto

Al hacer clic en el menú *Planificar* el sistema mostrará la pantalla de la figura 13. Esta pantalla muestra un listado de las iteraciones del proyecto elegido en la pantalla 3. Bajo el nombre de cada iteración hay dos opciones, una que dice *Ver Tareas* y la otra *Modificar/Eliminar*. Al hacer clic en la segunda opción se mostrará la pantalla de la figura 14, en la cual puede modificar o eliminar la iteración.

Figura 13: Listado de iteraciones del proyecto actual.

Por otra parte con la opción *Ver Tareas* se mostrará la pantalla de la figura 15. En esta pantalla se listan las tareas agrupadas por disciplina. Se muestran las fechas de inicio de la tarea, el tiempo estimado para su finalización y el porcentaje de avance. Del lado derecho se muestra el diagrama de Gannt correspondiente a la programación de las tareas.

En la pantalla 15, al hacer clic sobre una tarea se presentará una pantalla donde se puede modificar información asociada a la tarea. Ver figura 16.

Después de guardar los cambios se presentará una pantalla de confirmación, en donde se tiene la opción *Regresar*, con la cual volverá la pantalla 15, mostrando el diagrama de Gannt actualizado.

Por otra parte, en la pantalla 13 de lado derecho tiene las opciones *Nueva Iteracion* y *Cambiar de proyecto actual*. Con la primera opción se muestra la pantalla 17, desde la cual se puede registrar una iteración.

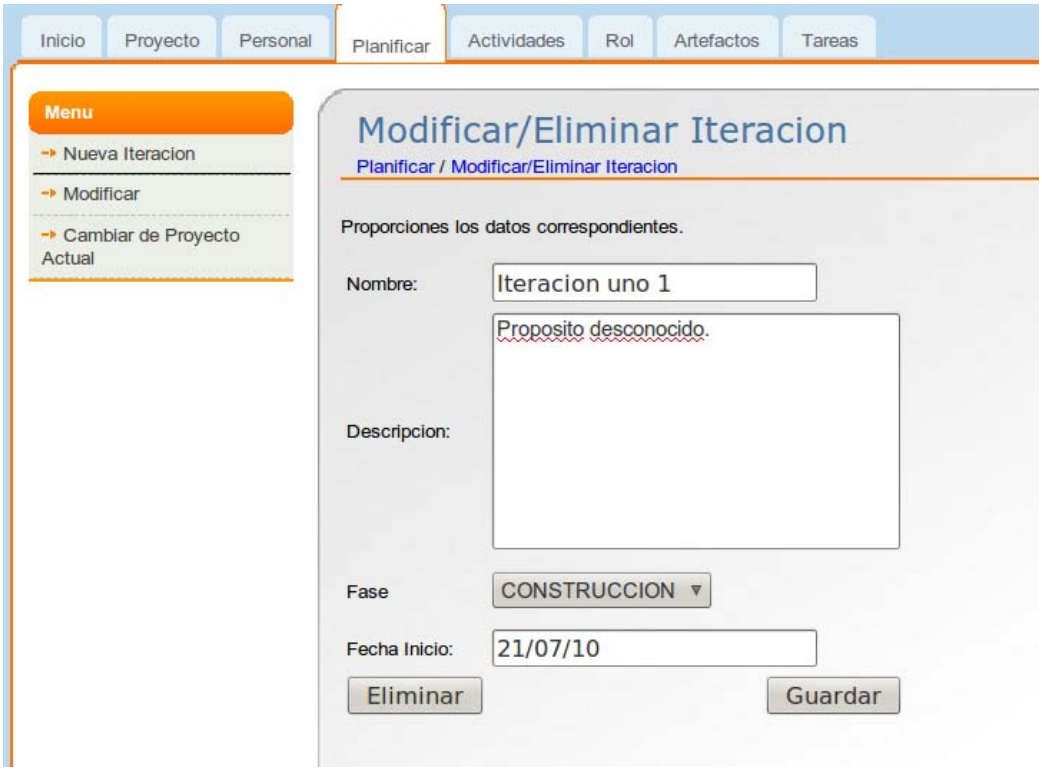

Figura 14: Pantalla para modificación o eliminación de la iteracion.

Por último la opción *Cambiar de proyecto actual* lleva a la pantalla 3, en la cual podemos cambiar de proyecto.

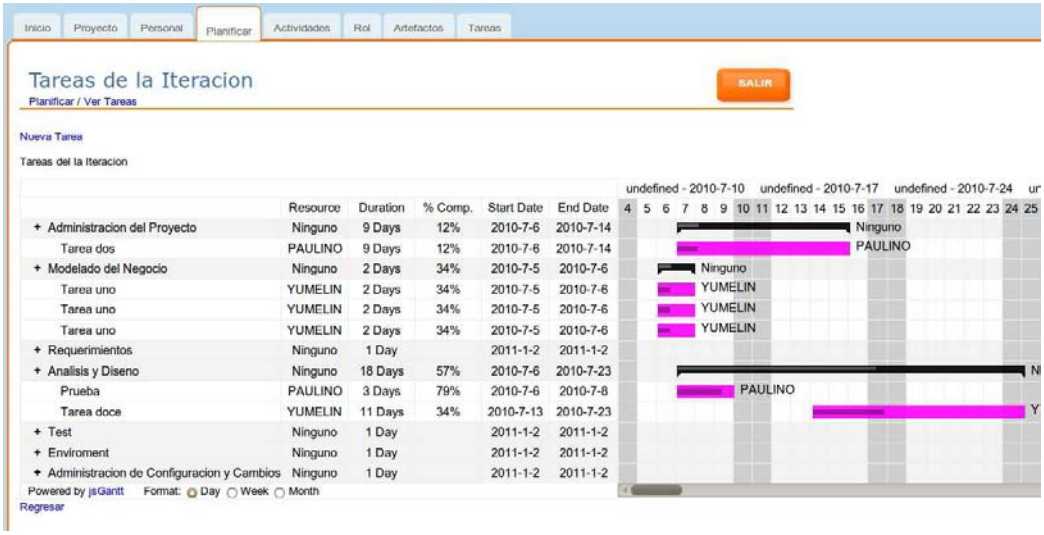

Figura 15: Listado de tareas de la iteración y diagramas de Gannt.

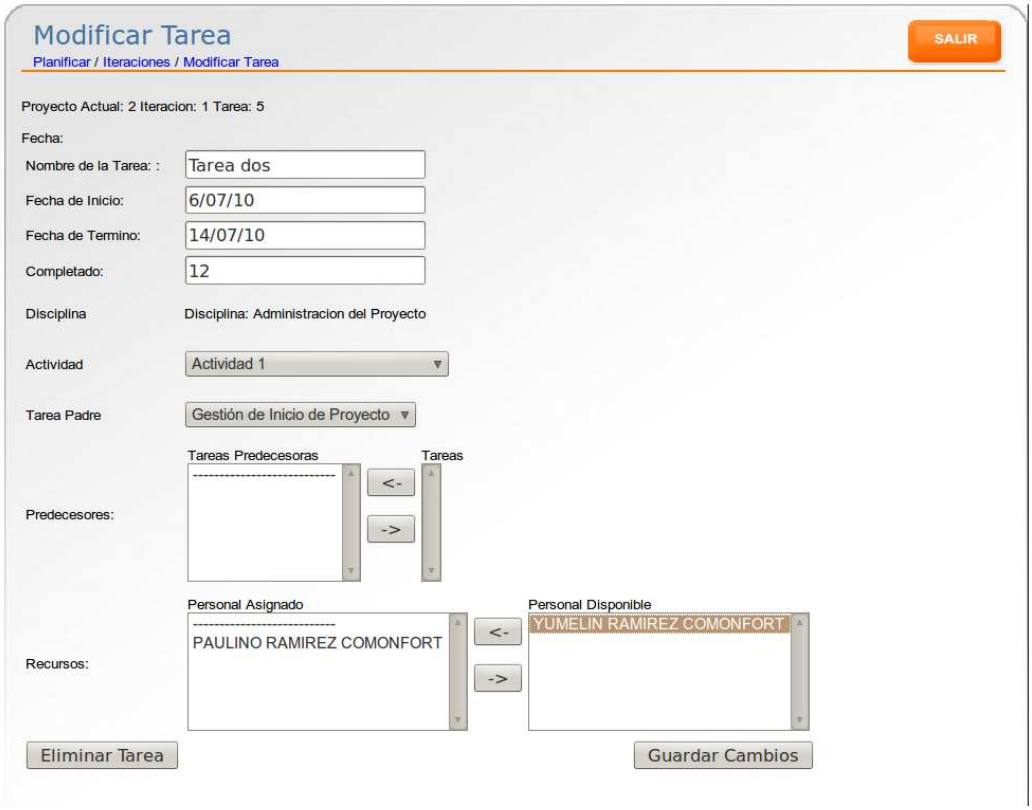

Figura 16: Pantalla para modificación de la tarea.

Figura 17: Pantalla para registro de una iteración.

0.5. Actividades

En este menú la pantalla que se presenta muestra una lista de todas las actividades dadas de alta en el sistema. Ver la pantalla de la figura 18. Se muestra el nombre de la actividad, el rol de quien puede efectuar la actividad y un enlace para modificar o eliminar la actividad. Al hacer clic sobre el nombre de la actividad se presentará una pantalla que muestra información detallada de la actividad, la pantalla es del estilo de la mostrada en la figura 19. En dicha pantalla se muestra el nombre de la actividad, los roles asociados, la disciplina a la que pertenece, los artefactos requeridos y los producidos.

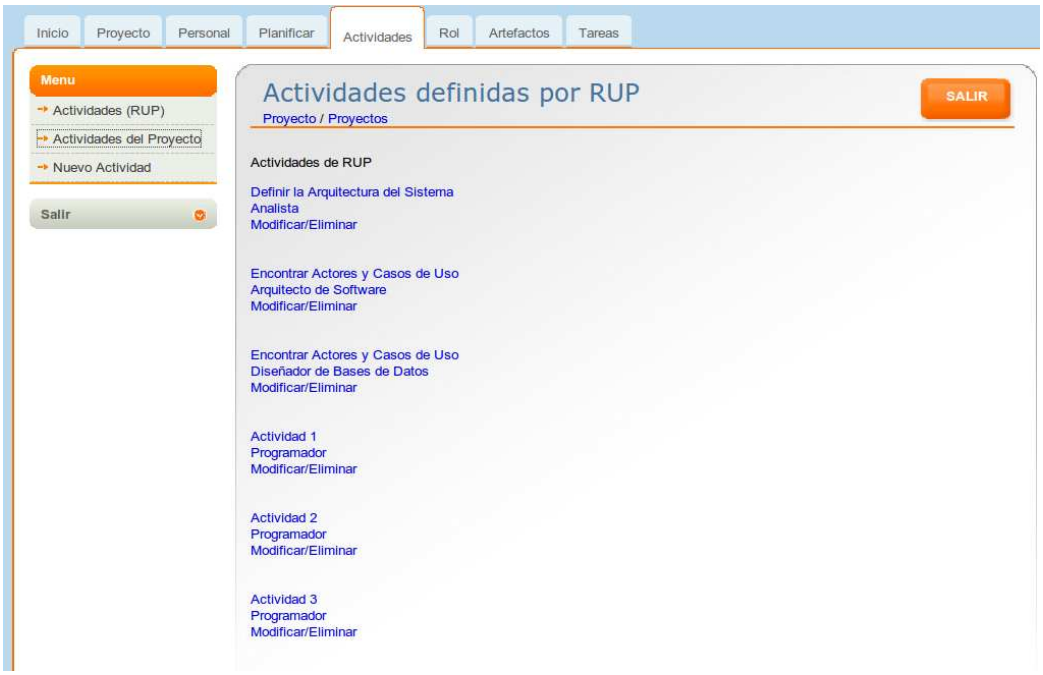

Figura 18: Listado de Actividades.

También al hacer clic sobre el nombre del rol en la pantalla 18, se presenta información detallada del rol, tal como se ve en la figura 20.

Por otra parte, para poder modificar una actividad hay que hacer clic sobre el enlace *Modificar/Eliminar*, esta opción despliega la pantalla de la figura 21. Contiene campos para escribir el propósito o información general, también dos listas para asignar los artefactos de entrada y producidos por la actividad.

Por último para agregar una actividad hay que hacer clic en el submenú del lado izquierdo que dice *Nueva Actividad* con lo cual aparecerá la pantalla de la figura 22.

Figura 19: Detalles de una actividad.

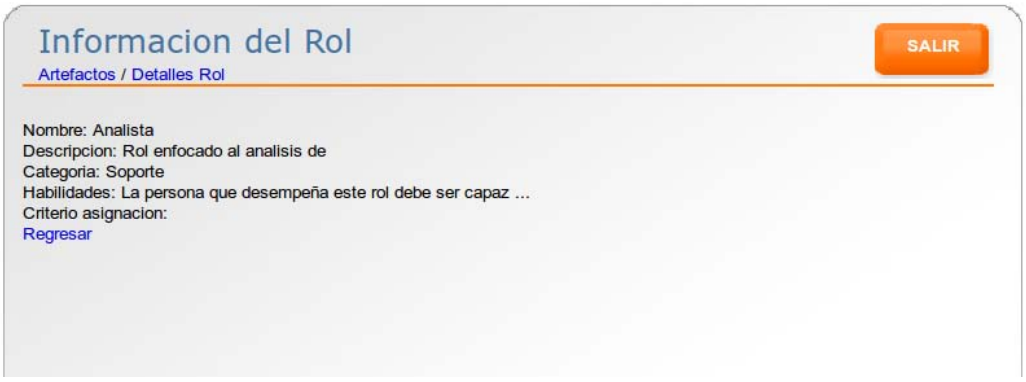

Figura 20: Detalles de un rol.

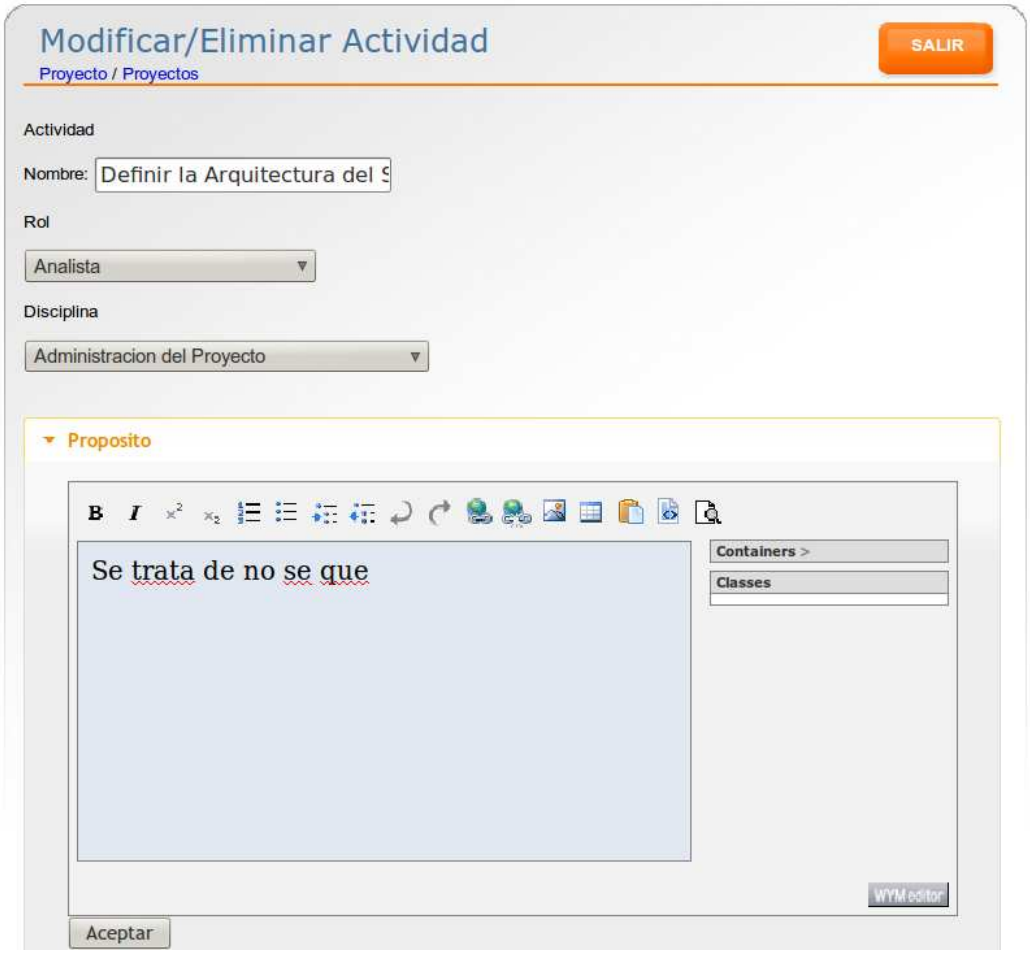

Figura 21: Modificación de una actividad.

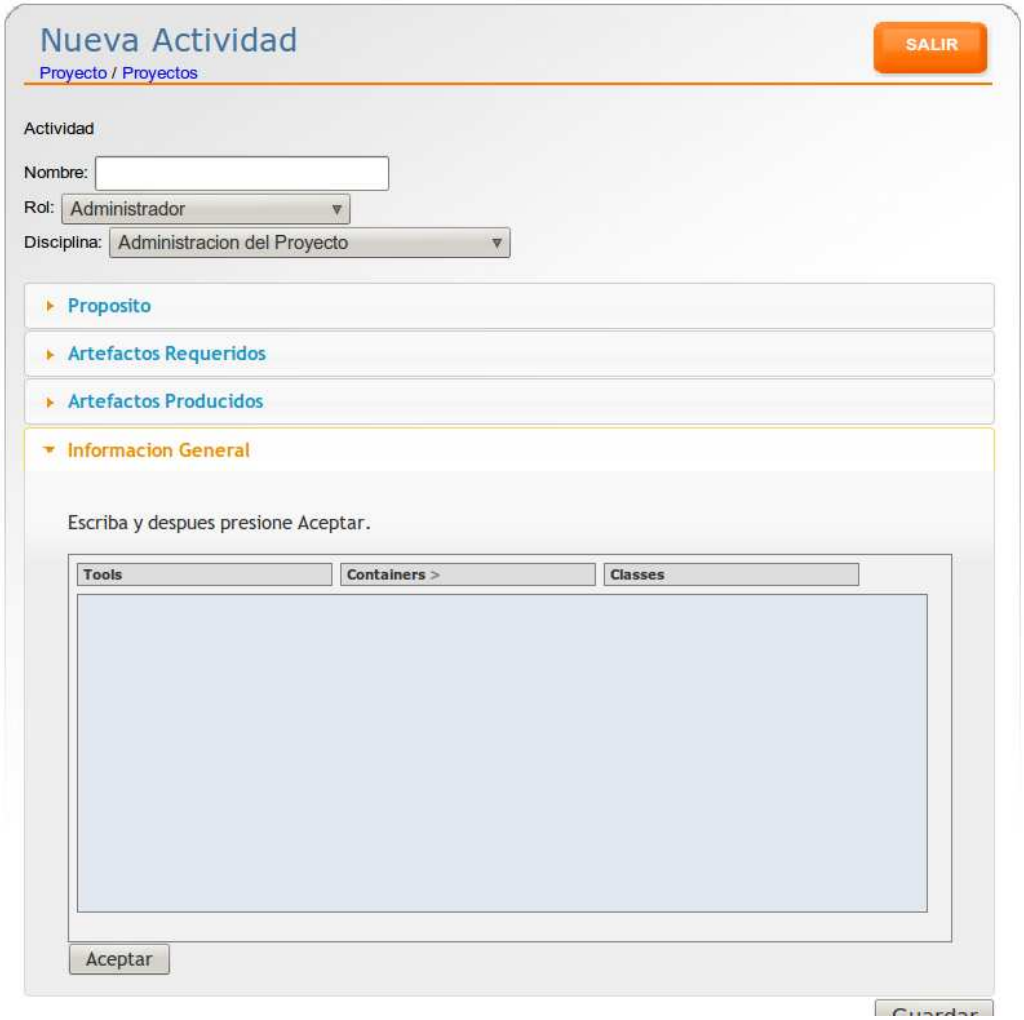

Figura 22: Pantalla para agregar una nueva actividad.

0.6. Administración de Roles

Para la administración de roles hay que hacer clic en el menú *Rol*, tras lo cual se mostrará la pantalla de la figura 23. Desde esta pantalla al hacer clic sobre *Modificar/Eliminar Rol* se muestra la pantalla de la figura 24. Desde la cual puede hacer modificaciones y posteriormente guardar los cambios.

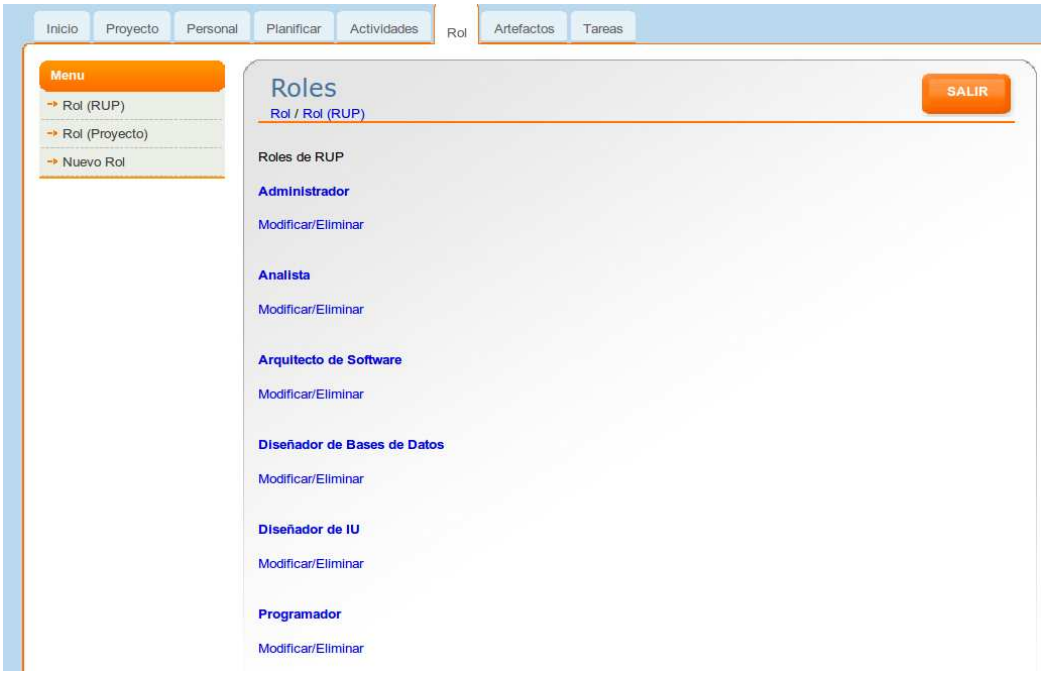

Figura 23: Listado de roles registrados en el sistema.

Además, desde la pantalla 23, al hacer clic sobre el nombre de un rol, se despliega una pantalla con la información detallada de dicho rol, similar al de la figura 20.

Para agregar un nuevo rol hay que hacer clic en la opción *Nuevo Rol* del submenú izquierdo de la pantalla. Después se mostrará la pantalla de la figura 25, en donde tiene que proporcionar los datos solicitados y después guardar.

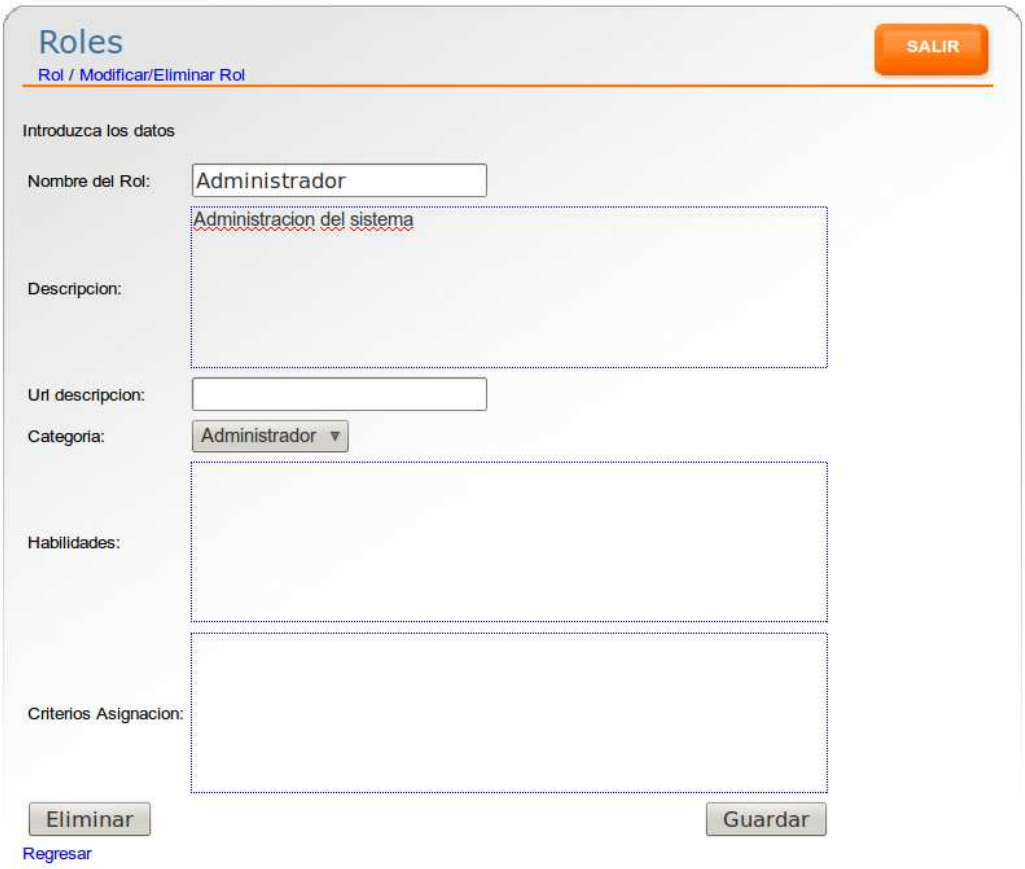

Figura 24: Pantalla para modificar o eliminar un rol.

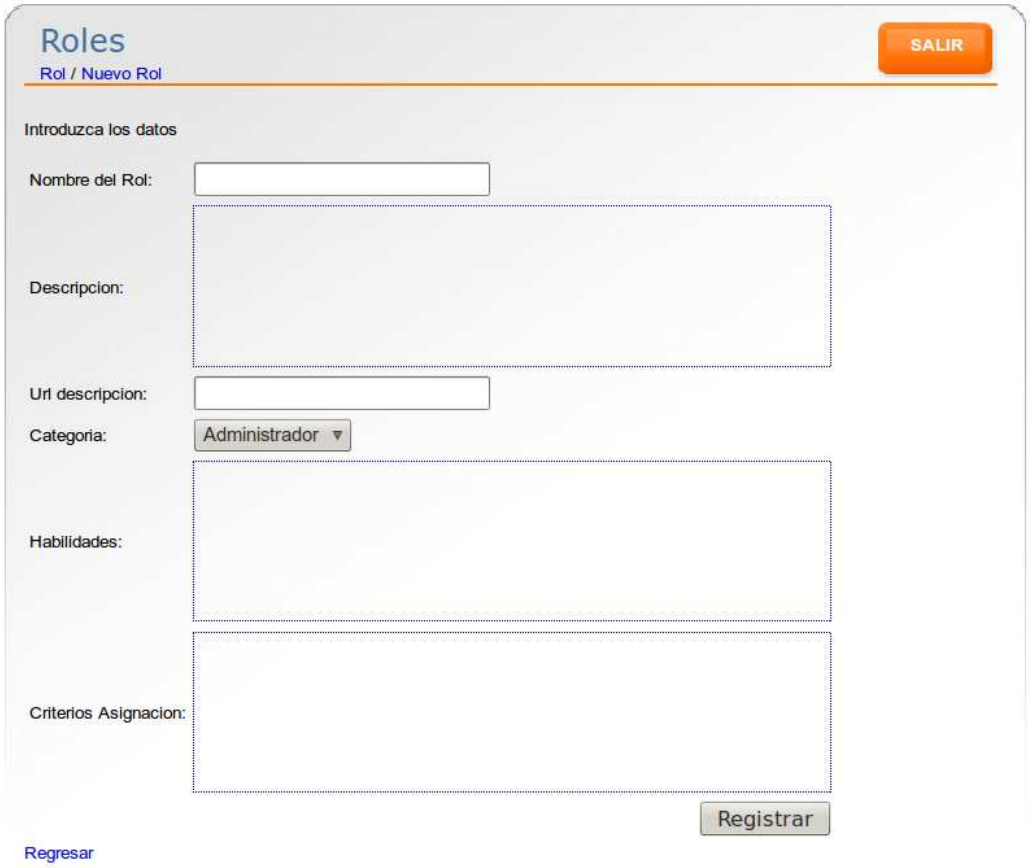

Figura 25: Pantalla para agregar un nuevo rol.

0.7. Administración de Artefactos

Para el manejo de los artefactos hay que hacer clic en el menú *Artefactos*, el cual nos manda la pantalla de la figura 26. Esta pantalla muestra una lista de los artefactos dados de alta en el sistema. Al hacer clic sobre el nombre de un artefacto, se despliega una pantalla mostrando información detallada del artefacto, similar al presentado en la figura 27.

Figura 26: Listado de artefactos del sistema.

Por otro lado, para la modificación o eliminación de un artefacto, hay que hacer clic sobre el enlace *Modificar/Eliminar*, tras lo cual se presentará la pantalla de la figura 28. Aquí se puede modificar toda la información asociada al artefacto elegido.

El registro de un nuevo artefacto es posible en la opción *Nuevo Artefacto* ubicada en el lado derecho de la pantalla. La pantalla de captura de datos se muestra en la figura 29.

Figura 27: Detalles de un artefacto.

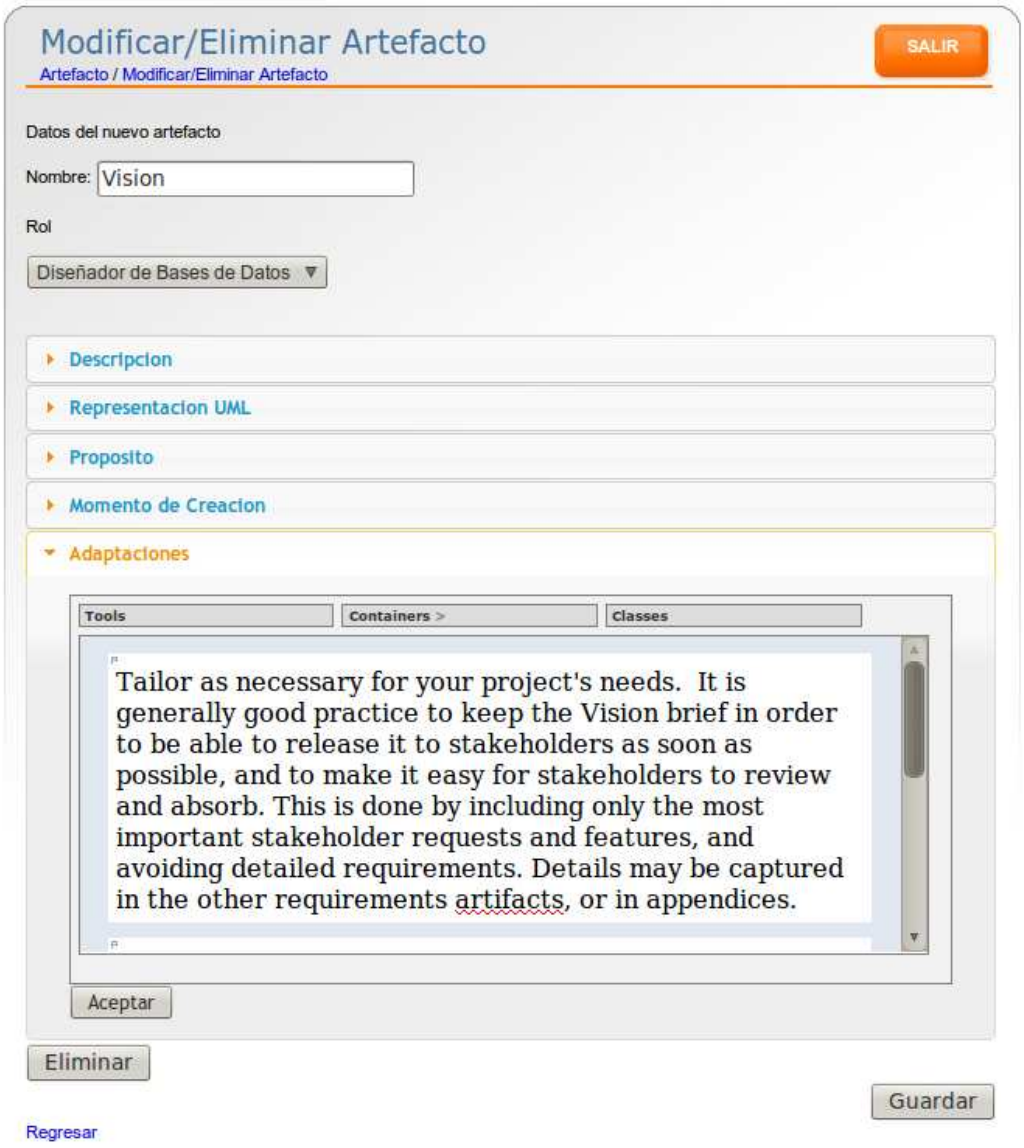

Figura 28: Modificación o eliminación de un artefacto.

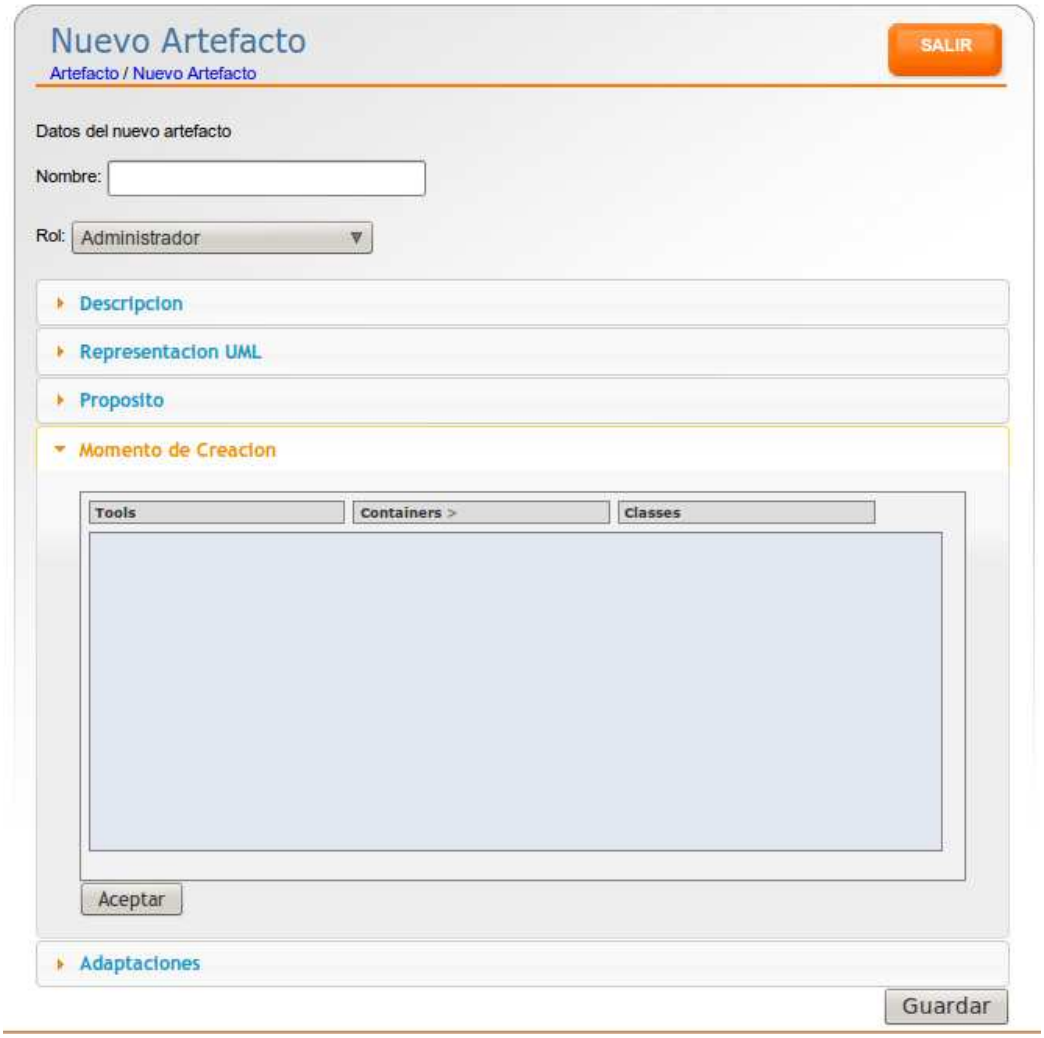

Figura 29: Pantalla para agregar un nuevo artefacto.

0.8. Administración de Tareas

Para gestionar las tareas asignadas a un usuario, hacer clic en el menú *Tareas*, se mostrará una pantalla con el listado de tareas asignadas al usuario, similar al de la figura 30.

Figura 30: Listado de tareas.

Al hacer clic sobre *Artefactos* se mostrará una lista de artefactos requeridos y producidos por la tarea, esto en base a la actividad que se tenga que realizar en la tarea. La pantalla se muestra en la figura 31.

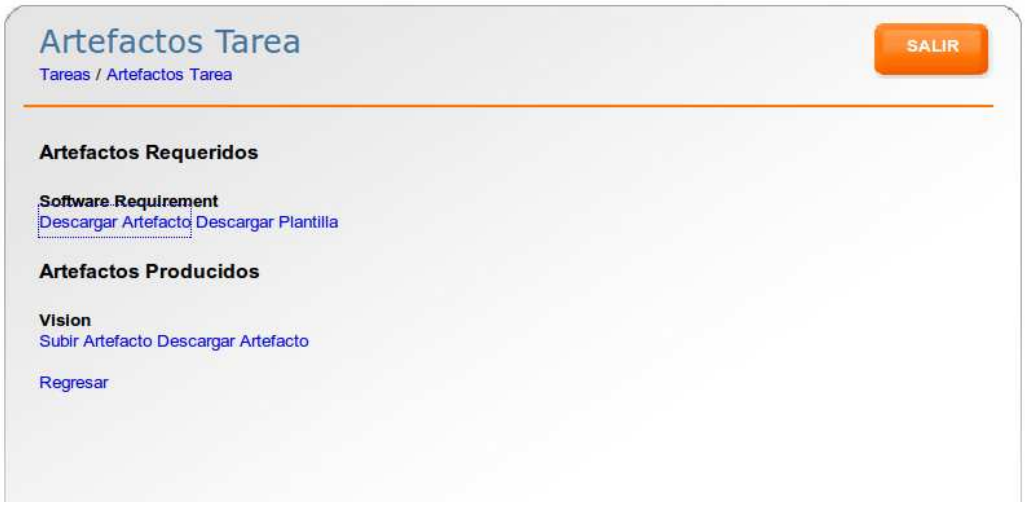

Figura 31: Listado de artefactos requeridos y producidos por la tarea.

En el listado de artefactos producidos, debajo del nombre de cada artefacto hay un enlace *Subir Artefacto*, el cual muestra la pantalla de la figura 32. Esta pantalla permite subir un archivo que será el ejemplar del artefacto para el proyecto actual.

Por otra parte si el artefacto ya tiene un ejemplar en el sistema, con la opción *Descargar Artefacto* nos permitirá descargar el archivo del servidor, tal como se ve

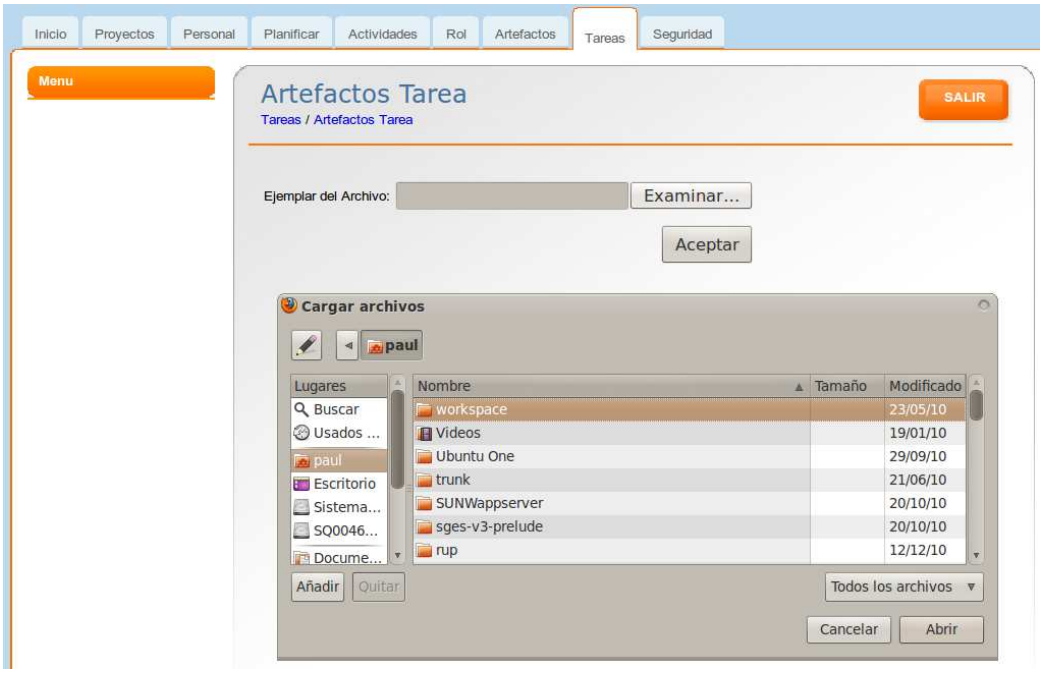

Figura 32: Pantalla para subir ejemplar de artefacto.

en 33.

Por último, desde la pantalla 30, al hacer clic en *Detalles* se muestra una pantalla con información resumida de la tarea, tal como se ve en 34.

Figura 33: Descargando un archivo del servidor.

Figura 34: Resumen de una tarea.

0.9. Seguridad

La gestión de privilegios en el sistema se puede realizar desde el menú *Seguridad*. Este presenta la pantalla de la figura 35. Desde esta pantalla hay que elegir un rol y posteriormente en *Aceptar* para que el sistema muestre los permisos que se tienen para dicho rol. Si se desea conceder o quitar el permiso para ejecutar o no una acción hay que hacer clic en *Cambiar*, tras lo cual el sistema realizará el cambio.

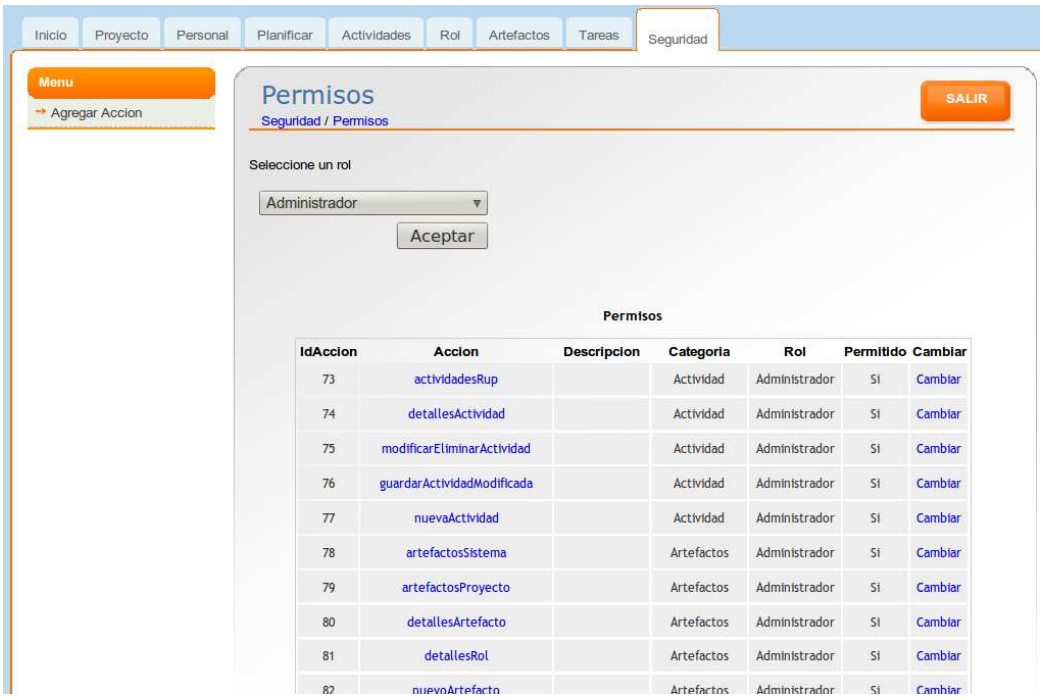

Figura 35: Pantalla para la gestión de permisos.

También al hacer clic sobre el nombre de una acción, se mostrará una pantalla para modificar dicha acción, similar al de la figura 36.

Finalmente, para agregar una acción para gestionar los privilegios en el sistema, hay que hacer clic en la opción *Agregar Accion* en el menú del lado izquierdo de la pantalla, con lo cual se presentará el formulario para capturar los datos necesarios. Ver figura 37.

Figura 36: Modificación o eliminación de una acción.

Figura 37: Pantalla para agregar una acción.

0.10. Instalación

Para poder operar el programa, primero se necesita contar con un contenedor se servlets, por ejemplo Tomcat. Este contenedor será el que desplegará la aplicación web. Para que lo anterior sea posible, es necesario proporcionar un archivo WAR[1] del proyecto al servidor. Para poder obtener el war es necesario crearlo con una de las muchas herramientas que existen. A continuación describiremos el proceso en el IDE Eclipse.

Primeramente tenemos que tener abierto el proyecto. Posteriormente clic derecho sobre la carpeta principal del proyecto, a continuación *Export...*→*Web*→*WAR File*, tras lo cual nos solicitará el nombre del archivo WAR a generar y la ruta destino.

Una vez generado el WAR sólo tenemos que copiarlo al directorio raiz de las aplicaciones web del servidor que se esté utilizando.

Así mismo es necesario crear la base de datos, el cual se crea desde el código sql proporcionado en la documentación.

[1] WAR o Web Archive es una especificación desarrollada por Sun que permite agrupar un conjunto de clases y documentos que conforman una aplicación Web en Java.