

**MANUAL TÉCNICO DEL SISTEMA DE DETECCION DE NECESIDADES DE CAPACITACIÓN – DNC-**

Elaborado por

Ing. Ludwin René Ayala Sánchez

## 

## HISTORIAL DE REVISIONES

|  |  |  |  |
| --- | --- | --- | --- |
| Fecha | Version | Descripción | Autor |
| 15/01/15 | 1.0 | Creación del documento | Ing. Ludwin Ayala |
|  |  |  |  |

**INTRODUCCIÓN**

El presente manual va dirigido al personal con conocimientos técnicos relacionados al área de informática, con el objetivo de proporcionar la lógica con la que se ha desarrollado el Sistema DNC. Aclarando que este manual no pretende ser un curso de aprendizaje de cada una de las herramientas empleadas para el desarrollo del sistema.

Para mayor detalle acerca de cada una de las herramientas utilizadas, y su forma de operación y aplicación, se recomienda consultar los manuales respectivos de cada una de ellos.

**OBJETIVO**

Proporcionar una guía para el lector, sobre el desarrollo de la interfaz y la funcionalidad del Módulo de DNC. Describir los componentes involucrados, utilidad y funcionamiento para el sistema, lo cual debe permitir una mejor visión a la hora de realizar un cambio o mejora al sistema.

**NIVELES O CAPAS**

Para la elaboración del módulo se ha considerado como modelo de arquitectura el J2EE (Java 2 Enterprise Edition), que reconoce hasta 7 niveles o capas, cada una con funcionalidades propias y diferenciadas pero no excluyentes que permiten manejar de forma más modular el desarrollo de cualquier aplicación, por ello se dice que se basa en el uso de componentes ampliamente modulares. Para el presente módulo se utilizarán 4 capas, que es mínima estructuración lógica que debería implementar una aplicación.



### CAPA DE PRESENTACIÓN

Presenta el sistema al usuario, comunica la información al usuario y captura la información proporcionada por el mismo.

### CAPA LÓGICA DEL NEGOCIO

Es donde residen los servicios que se ejecutan, se reciben las peticiones del usuario y se envían las respuestas tras el procesamiento correspondiente. Se denomina capa lógica del negocio pues es aquí donde se establecen todas las reglas que deben cumplirse.

### CAPA DE INTEGRACIÓN

Es donde se suministra una interfaz común entre la aplicación y uno o más dispositivos de almacenamiento de datos, tales como una Base de datos o un archivo. Su principal función es aislar una aplicación de la tecnología de persistencia Java subyacente (API de Persistencia Java), la cual podría ser JDBC, JDO, Enterprise JavaBeans, TopLink, EclipseLink, Hibernate, iBATIS, o cualquier otra tecnología de persistencia, lo que nos facilita su actualización o manipulación sin cambiar otras partes de la aplicación.

### CAPA DE PERSISTENCIA

También conocida como DAL es donde residen los datos y es la encargada de acceder a los datos. Está formada por los gestores de bases de datos que realizan todo el almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio.

**TECNOLOGÍAS DE DESARROLLO**

A continuación se lleva a cabo un breve detalle de cada una de las tecnologías utilizadas en el desarrollo del sistema.

### JSF + FACELETS

Java Server Faces es un estándar dentro de la pila JEE (Java Enterprise Edition) que permite el desarrollo de la capa de presentación y control para aplicaciones Web.

JSF permite el desarrollo de aplicaciones Web con base a componentes y eventos. Aproximación típica para el desarrollo de aplicaciones interactivas en entornos de escritorio. De esta manera abandona la aproximación tradicional y más limitada de "petición/respuesta", en la que se trabaja básicamente con documentos y apuesta por el trabajo con pantallas de interacción.

### PRIMEFACES

Primefaces es una librería de componentes para JSF. Estos componentes aportan, frente a los componentes estándar de JSF, una abstracción para el uso de la tecnología AJAX ya soportada en JSF 2. El desarrollador puede centrarse en la funcionalidad ofrecida sin tener que preocuparse del JavaScript que se ejecutará en el cliente o de que partes de la pantalla serán necesarias refrescar en respuesta de un evento en la interfaz de usuario.

No siendo Primefaces parte del estándar JEE, ahora es la única librería de componentes visuales que podemos decir que soporta de manera estable la versión 2 de JSF.

### JQUERY

JQuery es una librería JavaScript para conseguir una interfaz rica de usuario en aplicaciones Web (RIA - Rich Internet Application). Esto quiere decir que esta librería JavaScript nos permitirá dotar a las páginas Web de efectos visuales para asemejar su uso al de aplicaciones tradicionales de escritorio, mejorando en gran medida la usabilidad de las mismas.

### JAVA EE

Java Platform, Enterprise Edition o Java EE (anteriormente conocido como Java 2 Platform, Enterprise Edition o J2EE hasta la versión 1.4; traducido informalmente como Java Empresarial), es una plataforma de programación para desarrollar y ejecutar software de aplicaciones en el lenguaje de programación Java. Permite utilizar arquitecturas de N capas distribuidas y se apoya ampliamente en componentes de software modulares ejecutándose sobre un servidor de aplicaciones.

### JPA (HIBERNATE)

JPA (Java Persisence API) es un estándar de Java (JSR-220) que permite a los desarrolladores trabajar de forma sencilla con bases de datos relacionales. JPA permite la abstracción de la fuente de datos, permitiendo que el código sea portable entre distintos sistemas gestores de bases de datos: Oracle, DB2, MySQL, PostgresSQL, SQL Server.

Gracias a JPA el desarrollador se puede centrar en la lógica de negocio, olvidando los detalles de implementación la capa de acceso a datos. Esto permite realizar desarrollos más rápidos y más seguros.

Hibernate es un estándar de facto ampliamente aceptado y extendido en el mercado. Tanto que la implementación de referencia de JPA es el EntityManager de Hibernate.

### ENTERPRISE JAVABEANS (EJB)

Los EJB proporcionan un modelo de componentes distribuido estándar del lado del servidor. El objetivo de los EJB es dotar al programador de un modelo que le permita abstraerse de los problemas generales de una aplicación empresarial (concurrencia, transacciones, persistencia, seguridad, etc.) para centrarse en el desarrollo de la lógica de negocio en sí. El hecho de estar basado en componentes permite que éstos sean flexibles y sobre todo reutilizables.

Con la tecnología J2EE Enterprise JavaBeans es posible desarrollar componentes (Enterprise beans) con posibilidad de reutilizar y ensamblar en distintas aplicaciones de la institución.

**MAVEN**

Es una herramienta de línea de comandos, sin interface gráfica ni nada que se le parezca, que nos ayuda a crear los directorios de nuestro proyecto y con las tareas habituales que se realizan en él, como compilado, generar jar, documentación, distribuir, dependencias con otros jar, etc.

### REPOSITORIOS MAVEN

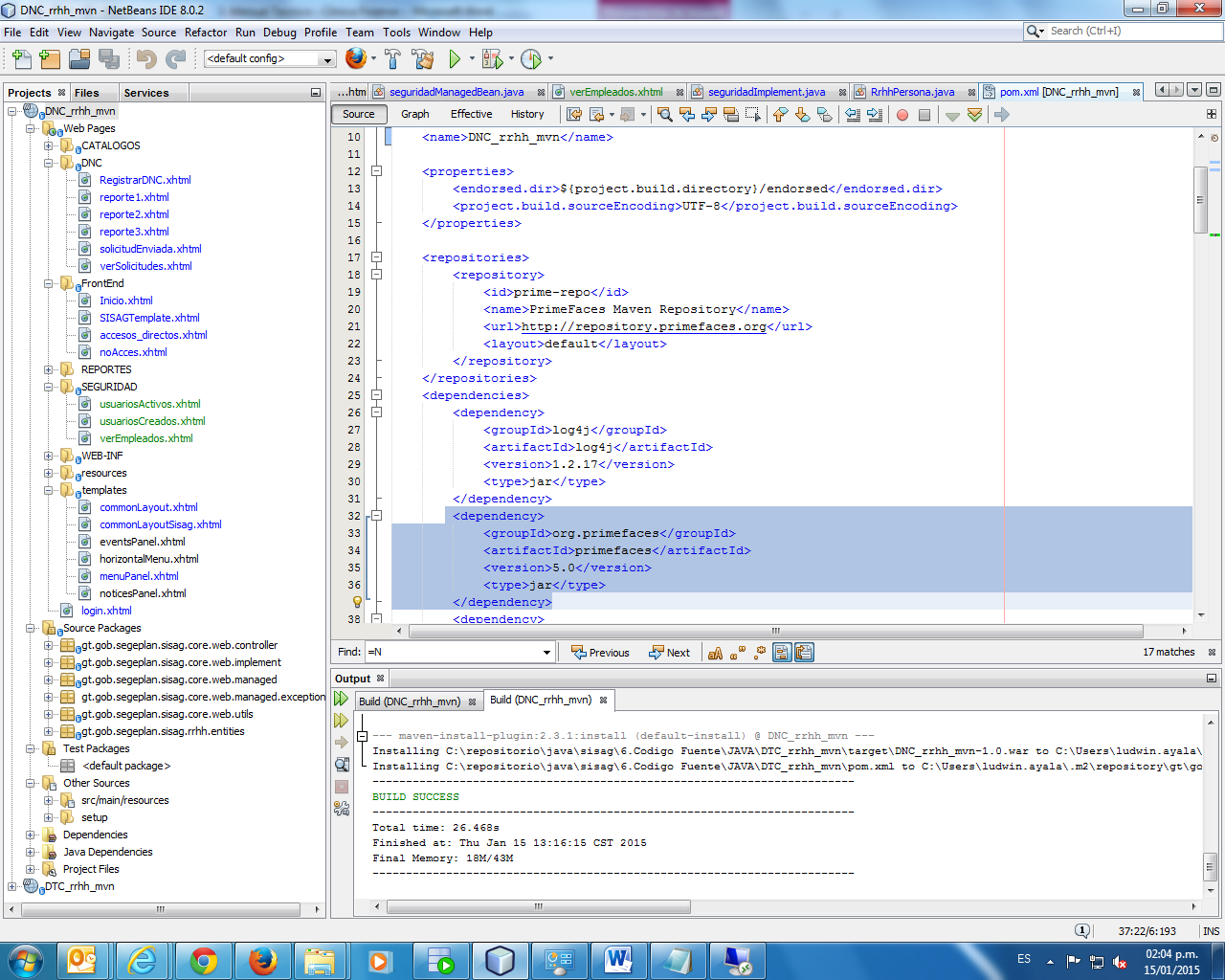
Una de las grandes ventajas de **MAVEN** son los repositorios (almacenes) de ficheros jar que se crea.

Todo lo que se baje **MAVEN** de internet o repositorio compartido lo almacena localmente, de forma que si lo necesita una segunda vez, no necesita descargárselo nuevamente de internet. Este directorio, se encuentra en el siguiente path:

* **C:\Documents and Settings\<USUARIO>\.m2** en Windows

### DEPENDENCIAS

Para agregar las dependencias que necesitamos como jar, es necesario editar el fichero **pom.xml**, de la siguiente forma:



En el siguiente ejemplo se muestra como se agrega una dependencia en el archivo de configuración, POM.XML.

<groupId>gt.gob.segeplan</groupId>

<artifactId>DNC\_rrhh\_mvn</artifactId>

<version>1.0</version>

<packaging>war</packaging>

<name>DNC\_rrhh\_mvn</name>

<properties>

<endorsed.dir>${project.build.directory}/endorsed</endorsed.dir>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

</properties>

<repositories>

<repository>

<id>prime-repo</id>

<name>PrimeFaces Maven Repository</name>

<url>http://repository.primefaces.org</url>

<layout>default</layout>

</repository>

</repositories>

<dependencies>

<dependency>

<groupId>log4j</groupId>

<artifactId>log4j</artifactId>

<version>1.2.17</version>

<type>jar</type>

</dependency>

**HERRAMIENTAS DE DESARROLLO**

Las herramientas utilizadas para el desarrollo son las siguientes:

- NetBeans IDE 8 (Entorno de Desarrollo Integrado).

- JDK 7 (Máquina Virtual Java).

- StarUML (Herramienta de UML).

- GlassFish 4(Servidor de aplicaciones).

- Oracle SQL Developer (Gestor de Base de Datos).

- PowerDesigner (Herramienta CASE).

- Chrome, Opera, IExplorer, Firefox (Pruebas de despliegue).

**BASE DE DATOS**

### NOMENCLATURA DE BASE DE DATOS

Los esquemas son los siguientes:

|  |  |  |
| --- | --- | --- |
| Nombre | Prefijo | Descripción |
| GENERAL | GEN | Almacena información de uso común. |
| SEGURIDAD | SEG | Este esquema contiene la estructura para el control de accesos, permisos y usuarios. |
| RECURSOS HUMANOS | RRHH | Contiene las tablas con información referente a Recursos Humano. |

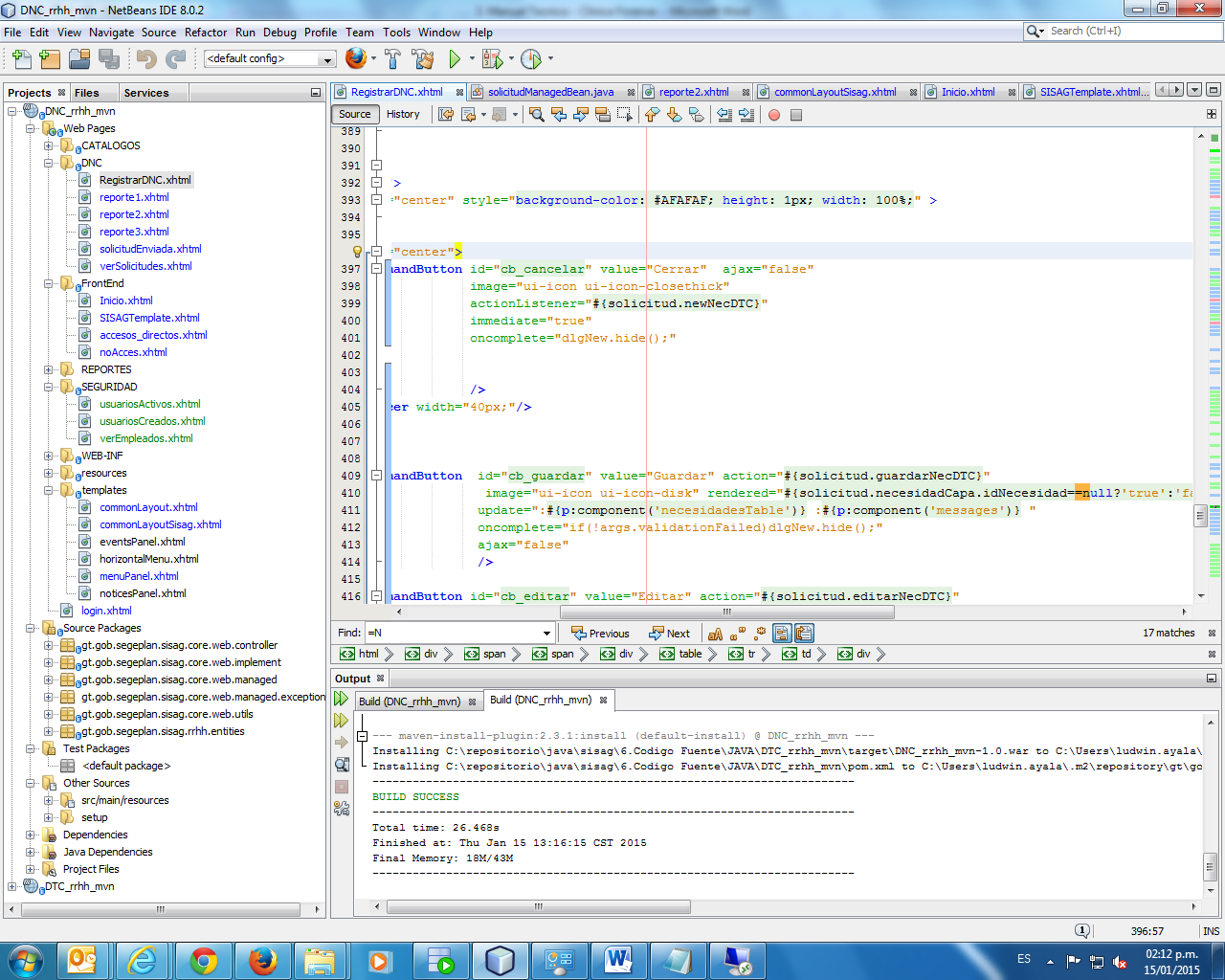
**LENGUAJE DE PROGRAMACIÓN**

**JAVA PLATFORM, ENTERPRISE EDITION** o **JAVA EE**

Es un lenguaje de programación orientado a objetos y desarrollado por Sun Microsystems.

Permitiendo utilizar arquitecturas de N capas distribuidas y se apoya ampliamente en componentes de software modulares ejecutándose sobre un servidor de aplicaciones.

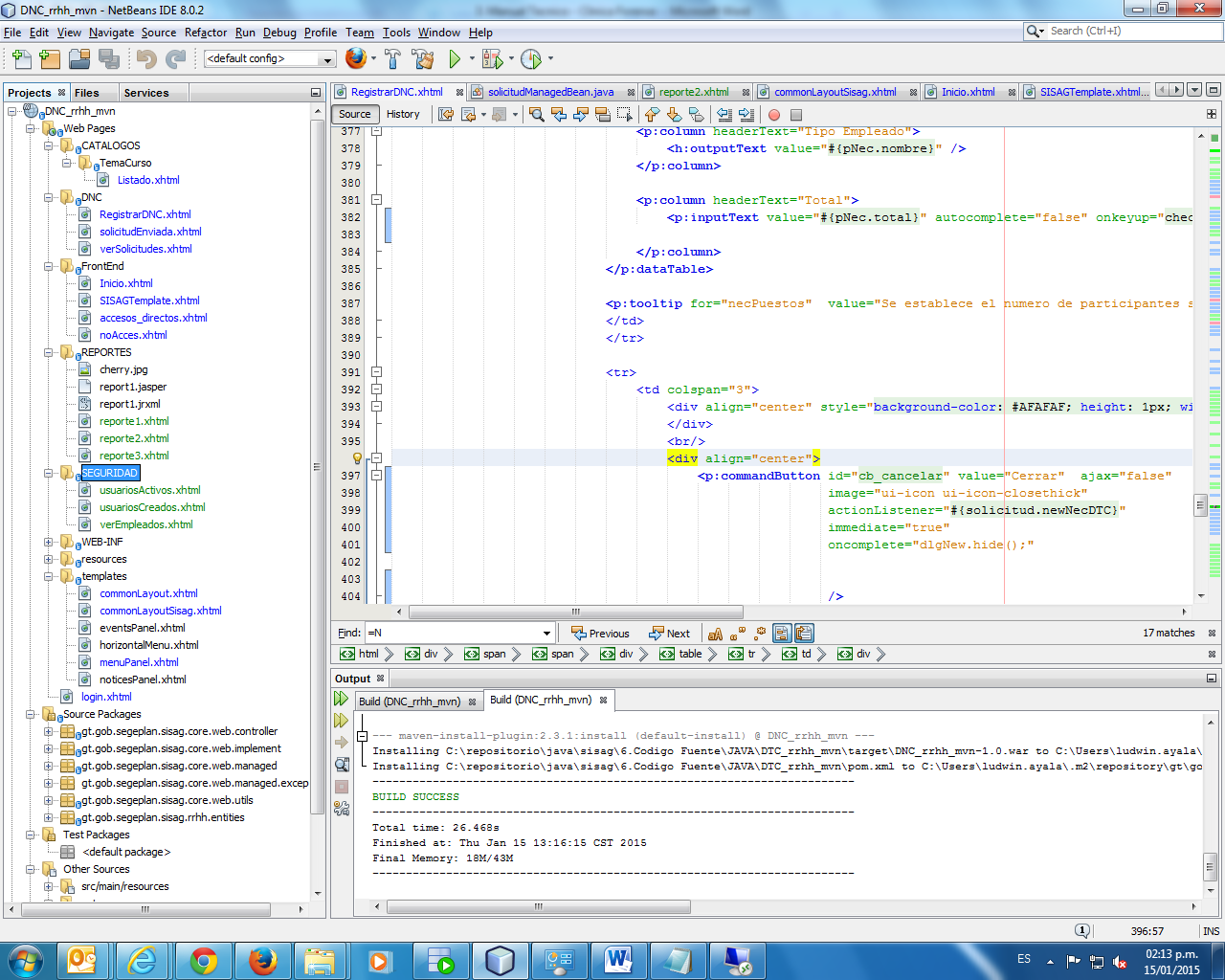
**ENTORNO DE DESARROLLO**



Se presenta la siguiente estructura de paquetes y directorios como están organizados los componentes desarrollados para el módulo de clínica forense.

**WEB PAGES**

(Directorio en donde se colocan páginas web y subdirectorios con páginas web).



**- CATALOGOS** (directorio en donde se colocan las páginas correspondientes al mantenimiento de catálogos).

**- DNC** (directorio en donde se colocan las páginas correspondientes al Detección de Necesidades de Capacitación).

**- FrontEnd** (directorio en donde se colocan las páginas correspondientes a la pantalla principal del sistema).

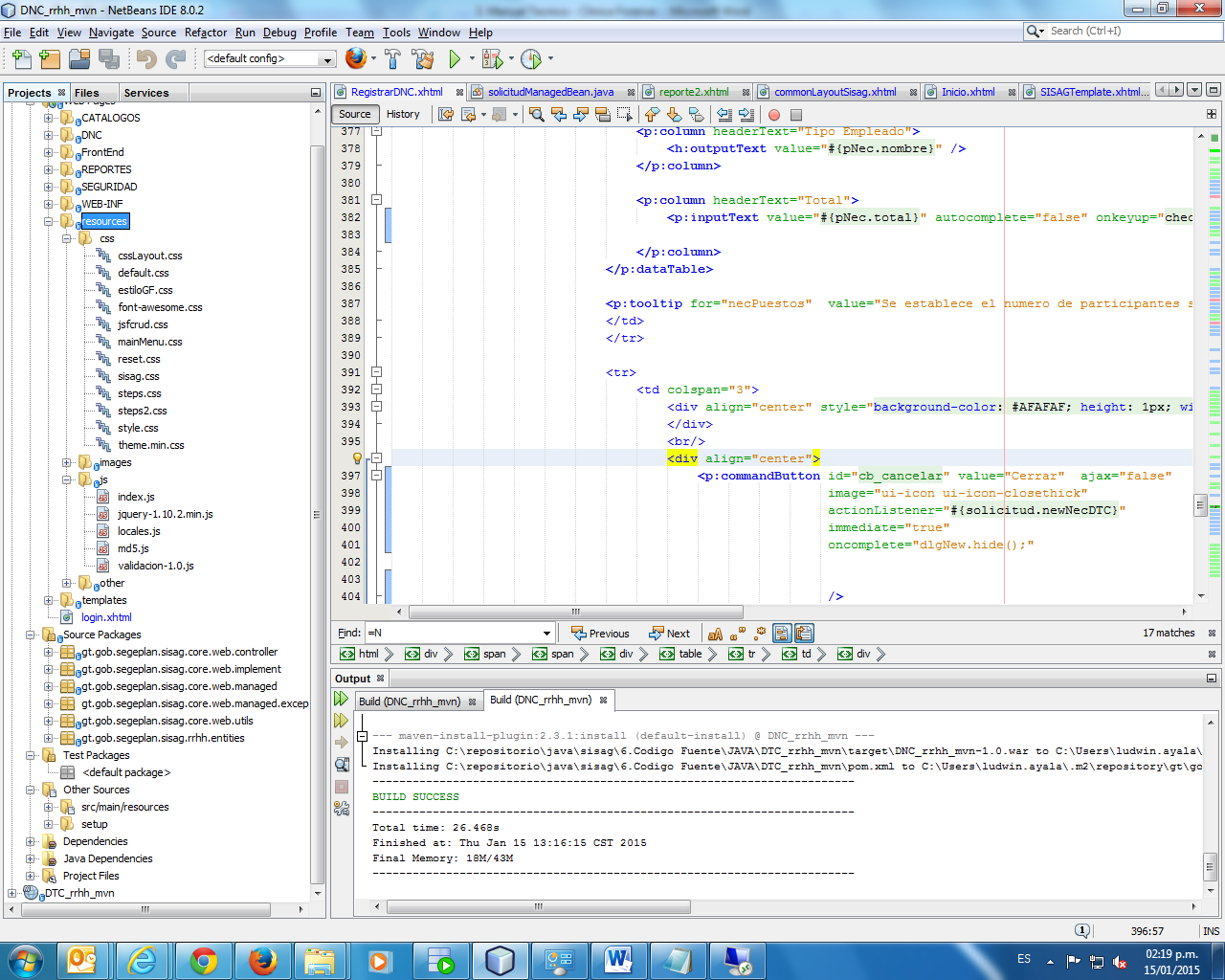
**- REPORTES** (directorio en donde se colocan las páginas correspondientes a los reportes generados).

**- SEGURIDAD** (directorio en donde se colocan las páginas correspondientes al mantenimiento de seguridad y accesos).

**-** **templates** (directorio en donde se colocan las páginas correspondientes los Templates, layouts y formatos).

**RESOURCES**

(Directorio en donde se colocan los recursos de look and feel, de formato y visualización).



* **css** (directorio en donde se colocan hojas de estilo css).
* **images** (directorio en donde se colocan iconos e

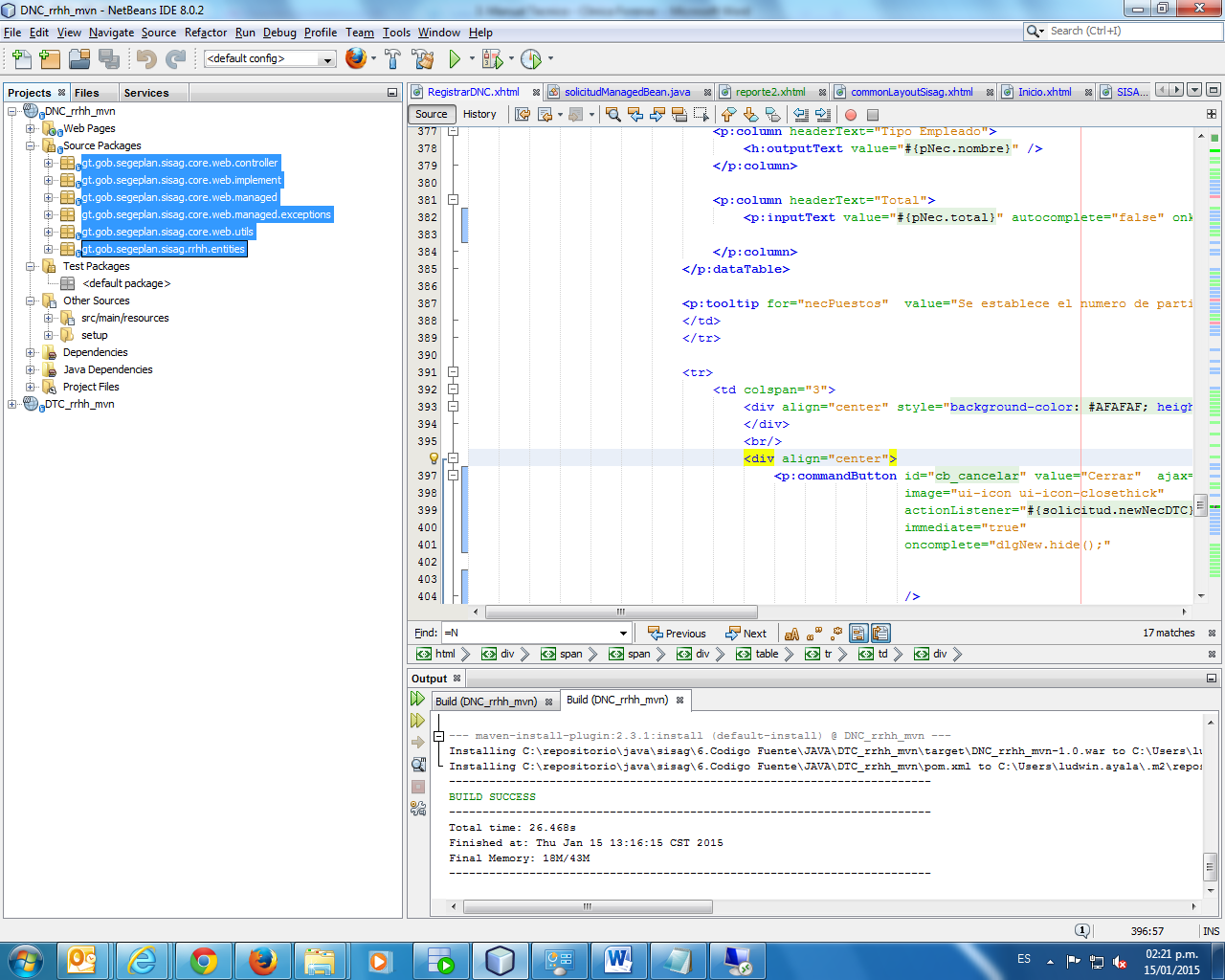
imágenes utilizadas por la aplicación).

* **js** (directorio donde se colocan librerías de java script).
* **other** (directorio donde se colocan archivos fuente

necesarios para el look and feel).

**SOURCE PACKAGES**

(Directorio con las fuentes de java).



### RESPONSABILIDADES DE CADA PAQUETE

|  |  |
| --- | --- |
| **Paquete** | **Responsabilidad** |
| gt.gob.segeplan.sisag.core.web.controller | Agrupa todas las clases que centralizan las peticiones entre la capa de negocio y acceso a datos, proporcionando el correcto flujo de control y poder realizar operaciones de persistencia. |
| gt.gob.segeplan.sisag.core.web.implement | En este paquete se almacenan todas las clases que utilicen la Unidad de Persistencia, la función de estas es el acceso a datos en sí. |
| gt.gob.segeplan.sisag.core.web.managed | En este paquete se almacenan todos los managed beans que se utilicen. Un manager debe contener todas las reglas del negocio para realizar una función en específico, por ejemplo para verificar si la solicitud ya ha sido enviada o no. |
| gt.gob.segeplan.sisag.core.web.utils | Agrupa todas las clases de utilería que exponen funciones auxiliares. Ejemplo de clases que se ubican en este paquete son validadores y convertidores, encriptación y filtros. |
| gt.gob.segeplan.sisag.rrhh.entities | Agrupa todos los beans de entidad (pojos decorados). Que son las clases mapeadas respecto a las tablas de la base de datos de nombre OSNIP. |

**DIRECTORIOS DE TRABAJO**

Todo el desarrollo deberá estar en el path C: \repositorio\sisag, a partir del cual se despliega en subcarpetas de acuerdo a la tecnología y metodología utilizada:

* **C: \repositorio\sisag\1.Requerimientos** - Requerimientos de desarrollo y/o mejora.
* **C: \repositorio\sisag\2.UML** – Modelado del sistema, Diagramas UML.
* **C: \repositorio\sisag\3.Base de Datos** – Script y archivos para generación de BD.
* **C: \repositorio\sisag\4.Codigo Fuente** – Archivos fuente del sistema páginas, css, js.
* **C: \repositorio\sisag\5.Documentos de Apoyo** – Documentos útiles para entender el sistema.
* **C: \repositorio\sisag\6.Manuales** – Documentación técnica del sistema.

Dentro de cada directorio se crea una referente al módulo al cual se hace referencia, como se muestra a continuación:

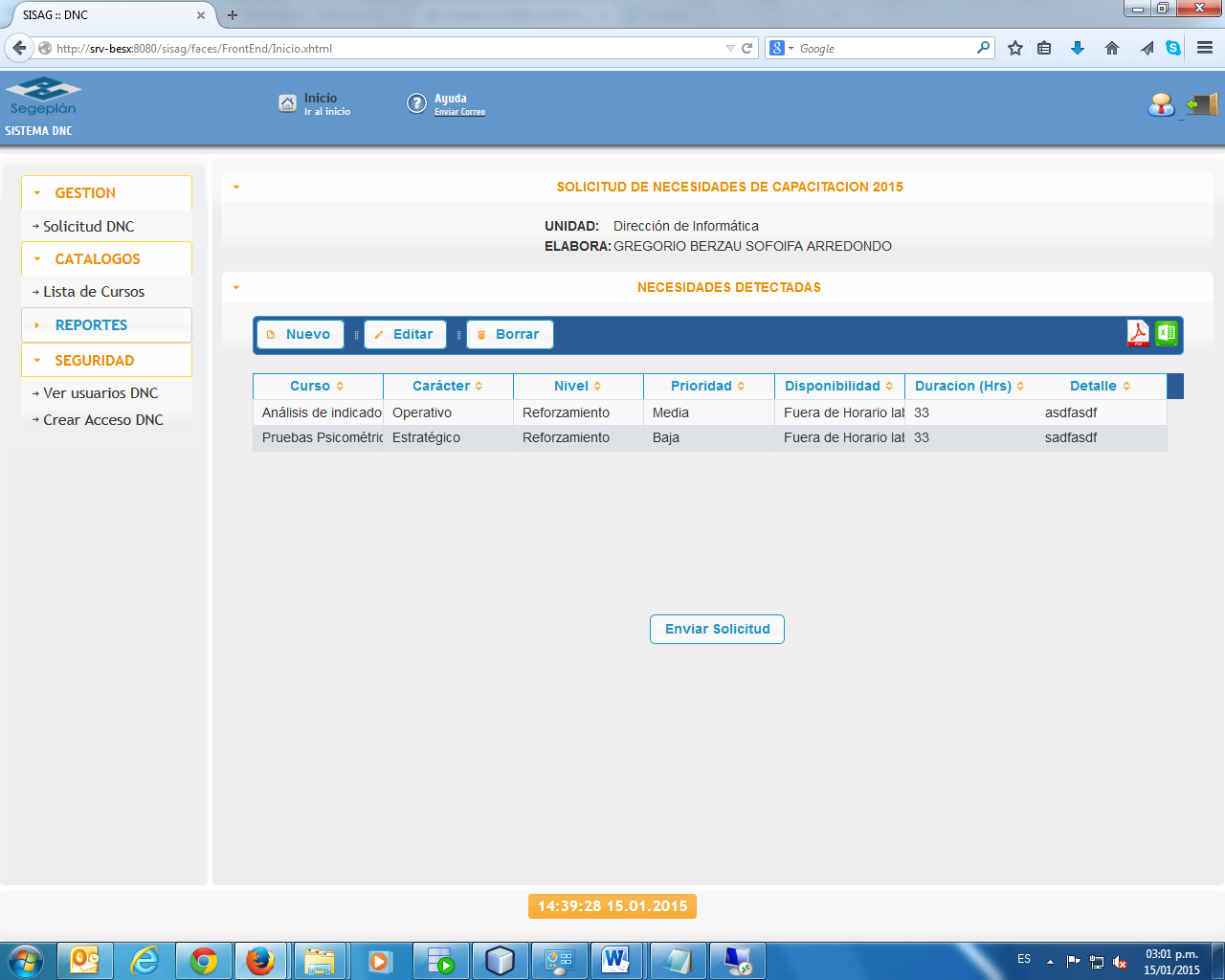
A partir del directorio del directorio de la tecnología y metodología:

**C: \repositorio\sisag\4.Codigo Fuente\DNC**

Las librerías utilizadas fuera del proyecto se descargan desde en el repositorio MAVEN, de acuerdo a la versión declarada en las dependencias en el archivo POM.

**PANTALLA PRINCIPAL**

A continuación se presenta la pantalla que establece la distribución y arquitectura de la información para las distintas páginas web del Sistema Informático de Administración y Gestión - SISAG, el cual cuenta con los siguientes elementos:



**1**

**2**

**3**

**4**

1. MENÚ DEL SISTEMA

Es un tipo de menú horizontal correspondiente a funciones globales del sistema.

1. MENÚ DE MÓDULOS DESPLEGABLE

Es un tipo de menú vertical correspondiente a las funciones de cada módulo a los cuales tenga acceso el usuario logeado en el sistema.

1. ÁREA DE TRABAJO

Es el área destinada a mostrar formularios e información al usuario. En esta sección se colocan todos los elementos con los que se interactúa directamente como cajas de texto, etiquetas, selección de opciones, reportes y otros.

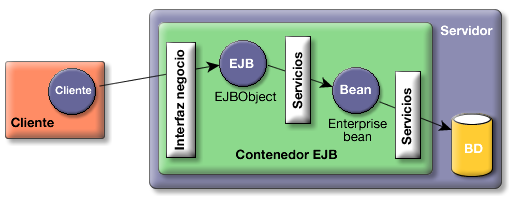
1. FOOTER

Información importante para el usuario.

**ANEXOS**

**FUNCIONAMIENTO DE EJB**

El funcionamiento de los componentes EJB se basa fundamentalmente en el trabajo del contenedor EJB. El contenedor EJB es un programa Java que corre en el servidor y que contiene todas las clases y objetos necesarios para el correcto funcionamiento de los Enterprise beans.



Representación de funcionamiento EJB.

La tecnología EJB define tres tipos de beans:

### BEANS DE ENTIDAD

Representan un objeto concreto que tiene existencia en alguna base de datos de la empresa. Una instancia de un bean de entidad representa una fila en una tabla de la base de datos.

### BEANS DIRIGIDOS POR MENSAJES

Estos pueden escuchar mensajes de un servicio de mensajes JMS. Los clientes de estos beans nunca los llaman directamente, sino que es necesario enviar un mensaje JMS para comunicarse con ellos.

### BEAN DE SESIÓN

Representa un proceso o una acción de negocio. Normalmente, cualquier llamada a un servicio del servidor debería comenzar con una llamada a un bean de sesión. Mientras que un bean de entidad representa una cosa que se puede representar con un nombre, al pensar en un bean de sesión deberías pensar en un verbo.