ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN

Titulación:

INGENIERO EN INFORMÁTICA

Título del proyecto:

CMS-JAVA

MEMORIA

Miguel Larraz Sala

Diego Pérez

Pamplona, Fecha de defensa

Control de cambios

Fecha	Autor	Descripción	Versión
22/12/2010	Miguel Larraz	Versión Inicial	0.1

Revisiones

Autor	Versión	Responsabilidad	Fecha

Índice de contenidos

1 VISIÓN Y ALCANCE.

1.1 Introducción.

El proyecto ha tenido como objetivo el desarrollo de un **sistema de gestión de contenidos.**

Un sistema de gestión de contenidos (CMS) consiste en una interfaz que controla una o varias bases de datos donde se aloja el contenido del sitio. El sistema permite manejar de manera independiente el contenido y el diseño. Así, es posible manejar el contenido y darle en cualquier momento un diseño distinto al sitio sin tener que darle formato al contenido de nuevo, además de permitir la fácil y controlada publicación en el sitio a varios editores. Un ejemplo clásico es el de editores que cargan el contenido al sistema y otro de nivel superior (directorio) que permite que estos contenidos sean visibles a todo el público.

Existen varios CMS actualmente, algunos ejemplos destacados son:

- **Vignette:** http://www.vignette.com/es

  Es un sistema CMS comercial. Fue el primer sistema CMS comercial que apareció en el mercado.

- **Drupal:** http://drupal.org/

  Uno de los CMS más populares, en este caso gratuito y open source. Creado en PHP y con posibilidad de utilizar varias bases de datos distintas, por defecto MySQL.

- **Mambo:** http://www.mamboserver.com/

  Un sistema CMS libre y gratuito, creado en PHP. Puedes leer el artículo sobre qué es Mambo publicado en DesarrolloWeb.com.

- **Joomla!:** http://www.joomla.org/

  Es un CMS de código libre, también creado en PHP. Surge como una mejora o ampliación de Mambo

- **Wordpress:** http://wordpress.org/

  El CMS para la creación de blogs por excelencia. Es el más utilizado y el mejor valorado, también creado en PHP y gratuito.

- **OsCommerce:** http://www.oscommerce.com/

  El sistema gestor de contenidos de código libre, para la creación de una tienda más conocido y utilizado.

1.2 Justificación.

Sin embargo, existe una carencia en estos CMS, no aprovechan las nuevas técnicas de desarrollo y la potencia de la tecnología JAVA EE6 y se ven limitados por plataformas tecnológicas más pobres. Esta tecnología proporciona un desarrollo rápido, dispone de muchos Frameworks que facilitan las tareas mas duras en el aspecto del desarrollo y permite unos desarrollos complejos, son altamente escalables y posee una integración completa con los protocolos standards usados en la red, independiza la aplicación de la plataforma y desde el punto de vista del negocio es una plataforma libre que no nos ata a ningún proveedor.

1.3 Objetivo // Requisitos.

Esta carencia ha motivado la realización del proyecto, que consiste en sistema de gestión de contenidos (CMS) modular, extensible, multipropósito y configurable gracias a los mas modernos standards de desarrollo JAVA.

Este CMS, permite la gestión integral de los contenidos y el formato de una WEB, sin necesidad de ningún conocimiento técnico, es completamente extendible, permitiendo instalar componentes como nuevos módulos con nuevas funcionalidades, nuevas plantillas, nuevos estilos, todo esto arropado por la potencia de la plataforma JAVA.

Las funcionalidades que el CMS soporta:

- Selección dinámica del flujo entre páginas.
- Agregar y quitar componentes.
- Cambio de estilos css.
- Cambio de plantillas del CMS.
- Colocación de componentes en diferentes posiciones.
- Sistema de instalación.
- Sistema de gestión de Usuarios.
- Inclusión de componentes básicos para tienda WEB.
- Soporte a componentes basados en JSF 2.0, javascript, HTML, xhtml y servlets.
- Interfaz de acceso a la base de datos para los componentes.
- Guardar Propiedades y objetos, accesibles entre componentes a nivel de sesión.
- Guardar Propiedades y objetos, accesibles entre componentes a nivel de aplicación.

1.4 Tecnologías.

1.4.1 JAVA EE6. (Link)

Java Platform, Enterprise Edition (Java EE) es el estándar industrial para el desarrollo de aplicaciones portátiles, robustas, escalables y seguras en el servidor. Basándose en la sólida base de Java SE, Java EE proporciona servicios Web, modelo de componentes, la gestión, y las API de comunicaciones que la convierten en el estándar de la industria para la aplicación de la arquitectura de clase empresarial orientada a servicios (SOA) y aplicaciones Web 2.0.

¿Cuáles son los principales beneficios de la plataforma Java EE?

La plataforma Java EE proporciona lo siguiente:

- Soporte completo de servicios Web. La plataforma Java EE proporciona un marco para desarrollar y desplegar servicios Web en la plataforma Java. El API Java para XML Web Services (JAX-WS) permite a los desarrolladores de tecnología Java desarrollar servicios SOAP interoperables y portátiles. Los desarrolladores usan el modelo estándar JAX-WS de programación para desarrollar clientes Web basados en SOAP de servicios y puntos finales. La parte cliente, se describe usando el Web Services Description Language (WSDL). JAX-WS permite a los clientes invocar los servicios Web, desarrollados a través de plataformas heterogéneas. De manera similar, JAX-WS la parte de servicio Web puede ser invocada por los clientes heterogéneos. Para obtener más información, consulte http://java.sun.com/webservices/.
- Soluciones más rápidas en cuanto al tiempo de entrega al cliente. La plataforma Java EE utiliza "contenedores" para simplificar el desarrollo. Los contenedores de Java EE prevén la separación de la lógica empresarial de gestión de recursos y del ciclo de vida, lo que significa que los desarrolladores pueden centrarse en escribir la lógica de negocio - su valor añadido - en lugar de escribir la infraestructura empresarial. Por ejemplo, el contenedor Enterprise JavaBeans (EJB) maneja la comunicación distribuida, ampliación, gestión de transacciones, etc Del mismo modo, Java Servlets simplifican el desarrollo Web, proporcionando la infraestructura para el componente, la comunicación y la administración de sesiones del contenedor Web que se integra con un servidor Web.
- La libertad de elección. La tecnología Java EE es un conjunto de normas que muchos vendedores pueden implementar. Los vendedores son libres de competir en las implementaciones, pero no en las normas o API. Sun proporciona una amplia compatibilidad de Java EE Test Suite (CTS) a los parnerts de Java EE. El Java EE CTS ayuda a garantizar la compatibilidad entre los proveedores de aplicaciones de servidor que facilita la portabilidad de las aplicaciones y los componentes escritos para la plataforma Java EE. La plataforma Java EE mantiene la filosofia Write Once, Run Anywhere (WORA) para servidores.
- Conectividad simplificada. La tecnología Java EE hace más fácil conectar las aplicaciones y sistemas que ya tiene y poner esas capacidades en la red, en los teléfonos y dispositivos. Java EE ofrece Java Message Service para la integración de diversas aplicaciones en un acoplamiento flexible, de forma asincrónica. La plataforma Java EE también ofrece soporte de CORBA para vincular estrechamente los sistemas a través de llamadas a métodos remotos. Además, la plataforma Java EE tiene conectores J2EE para ligarse a sistemas de información empresarial, como sistemas ERP, aplicaciones empaquetadas financieras, y las aplicaciones de CRM.
- Al ofrecer una plataforma con menor tiempo de entrega de la solución en el mercado, la libertad de elección, y conectividad simplificada, la plataforma Java EE ayuda a reducir el coste total de propiedad (TCO) y al mismo tiempo evitar una sola fuente de lock-in para sus necesidades de software empresarial.

1.4.2 Frameworks usados.

1.4.2.1 JSF 2.0 (link)

La tecnología JavaServer Faces incluye:

- Un conjunto de APIs para representar componentes de interfaz de usuario y la gestión de su estado, la gestión de eventos y validación de entrada, la definición de navegación de la página, y el apoyo a la internacionalización y accesibilidad.
- Un JavaServer Pages (JSP) biblioteca de etiquetas personalizadas para la expresión de una interfaz JavaServer Faces dentro de una página JSP.

Diseñado para ser flexible, la tecnología JavaServer Faces, estándar de interfaz de usuario basado en los conceptos de capa Web sin limitar a los desarrolladores un idioma determinado, protocolo o dispositivo cliente. Las clases de componentes de interfaz de usuario incluido con la tecnología JavaServer Faces encapsular la funcionalidad de los componentes, no la presentación específica del cliente, permitiendo así que los componentes JavaServer Faces UI ser prestados a los diferentes dispositivos cliente. Al combinar la funcionalidad de los componentes de interfaz de usuario con la extracción de datos personalizada y los atributos que definen la representación de un componente específico de la interfaz de usuario, los desarrolladores pueden construir etiquetas personalizadas para un dispositivo de cliente en particular. Para su comodidad, la tecnología JavaServer Faces proporciona un procesador personalizado y una biblioteca de etiquetas JSP personalizadas para representar a un cliente de HTML, permitiendo a los desarrolladores de Java Platform, Enterprise Edition (Java EE) utilizar la tecnología JavaServer Faces en sus aplicaciones.

La facilidad de uso es el objetivo principal, la arquitectura JavaServer Faces define claramente la separación entre la lógica de la aplicación y la presentación al tiempo que facilitan la conexión de la capa de presentación para el código de aplicación. Este diseño permite a cada miembro de un equipo de desarrollo de aplicaciones Web centrarse en su pieza del proceso de desarrollo, y también proporciona un modelo de programación sencillo para vincular las piezas. Por ejemplo, los desarrolladores de páginas Web sin conocimientos de programación pueden utilizar etiquetas de componente JavaServer Faces UI para ligarse a código de la aplicación desde una página Web, sin necesidad de escribir ningún script.

Desarrollado a través del Java Community Process bajo JSR - 314, la tecnología JavaServer Faces establece el estándar para la creación de interfaces del lado del servidor del usuario. Con las aportaciones del grupo de expertos, la API JavaServer Faces se están diseñando para que puedan ser aprovechados por herramientas que garanticen el desarrollo de aplicaciones Web sea aún más fácil. Varios proveedores de herramientas respetados eran miembros del grupo de expertos JSR-314, que desarrolló el JavaServer Faces 1.0. Estos proveedores se han comprometido a apoyar la tecnología JavaServer Faces en sus herramientas, fomentando así la adopción de la tecnología estándar JavaServer Faces.

1.4.2.2 EclipseLink (JPA 2.0) (Link)

Java Persistence API, más conocida por sus siglas JPA, es la API de persistencia desarrollada para la plataforma Java EE e La Java Persistence API, a veces referida como JPA, es un framework del lenguaje de programación Java que maneja datos relacionales en aplicaciones usando la Plataforma Java en sus ediciones Standard (Java SE) y Enterprise (Java EE).

La JPA fue originada a partir del trabajo del JSR 220 Expert Group. Ha sido incluida en el estándar EJB3.

La persistencia en este contexto cubre tres áreas:

- La API en sí misma, definida en javax.persistence.package
- La Java Persistence Query Language (JPQL)
- Metadatos objeto/relacional

El objetivo que persigue el diseño de esta API es no perder las ventajas de la orientación a objetos al interactuar con una base de datos (siguiendo el patrón de mapeo objeto-relacional), como sí pasaba con EJB2, y permitir usar objetos regulares (conocidos como POJOs).

1.4.2.3 EJB 3.1 (Link)

Introducción
La versión 3.1 de la especificación Enterprise JavaBeans (EJB) se simplifica y proporciona aspectos declarativos, tales como transacciones, seguridad y procesamiento asincrónico. Añade contextos e inyección de dependencias (CDI) que le dotan de más potencia y flexibilidad. Algunas de las características más destacables de EJB 3.1 son:

- Enterprise JavaBeans es ligero: La especificación EJB 3.1 es de peso ligero siguiendo el modelo de programación: Plain Old Java Object (POJO).
- Uso simplificado: El verdadero beneficio de EJB 3.1 es la definición declarativa de aspectos transversales.
- Transnacionalidad.
- Gestión de la concurrencia
- Inyección que permite la comunicación entre los diferentes EJB y tambien los beans y JSF.

1.4.3 Infraestructura.

1.4.3.1 MySQL (link)

La base de datos MySQL se ha convertido en la base de datos de código abierto más popular debido a su alto rendimiento, alta fiabilidad y facilidad de uso. También es la base de datos de elección para una nueva generación de aplicaciones basadas en la pila LAMP (Linux, Apache, MySQL, PHP / Perl / Python.) Muchos de los más grandes y las organizaciones de más rápido crecimiento del mundo, incluyendo Facebook, Google, Adobe, Alcatel Lucent y Zappos se basan en MySQL para ahorrar tiempo y dinero en sus grandes volúmenes de sitios Web, los sistemas críticos de negocio y paquetes de software.

MySQL se ejecuta en más de 20 plataformas, incluyendo Linux, Windows, Mac OS, Solaris, HP-UX, IBM AIX, que le da el tipo de flexibilidad que le da el control. Si eres nuevo en la tecnología de base de datos o un desarrollador experimentado o DBA, MySQL ofrece una amplia gama de herramientas de base de datos, asistencia técnica, capacitación y servicios de consultoría para que usted tenga éxito.

1.4.3.2 Glassfish 3.0.1 (link)

GlassFish es un servidor de aplicaciones de software libre desarrollado por Sun Microsystems, compañía adquirida por Oracle Corporation, que implementa las tecnologías definidas en la plataforma Java EE y permite ejecutar aplicaciones que siguen esta especificación. La versión comercial es denominada Oracle GlassFish Enterprise Server (antes Sun GlassFish Enterprise Server) . Es gratuito y de código libre, se distribuye bajo un licenciamiento dual a través de la licencia CDDL y la GNU GPL.

GlassFish está basado en el código fuente donado por Sun y Oracle Corporation, éste último proporcionó el módulo de persistencia TopLink. GlassFish tiene como base al servidor Sun Java System Application Server de Oracle Corporation, un derivado de Apache Tomcat, y que usa un componente adicional llamado Grizzly que usa Java NIO para escalabilidad y velocidad.

1.4.4 Tecnologías compatibles.

1.4.4.1 Introducción.

Este CMS, tiene como particularidad, admitir la instalación de componentes que usen diferentes tecnologías, a continuación enumeramos las tecnologías que el CMS admite para el desarrollo de los componentes.

1.4.4.2 Capa de presentación.

- **JSF 2.0.**
- **HTML.**
- **JSP.**

1.4.4.3 Capa de negocio.

- **Servlets.**
- **ManagedBeans.**

1.4.4.4 Otras.

En realidad se pueden desarrollar servlets también con el resto de tecnologías admitidas por el servidor de aplicaciones que estemos usando, aunque hay que tener en cuenta, que no siempre podrán acceder a los ManagedBeans que manejan el acceso a la capa de negocio propia del CMS, ni a los EJB que manejan el acceso a la capa de datos.

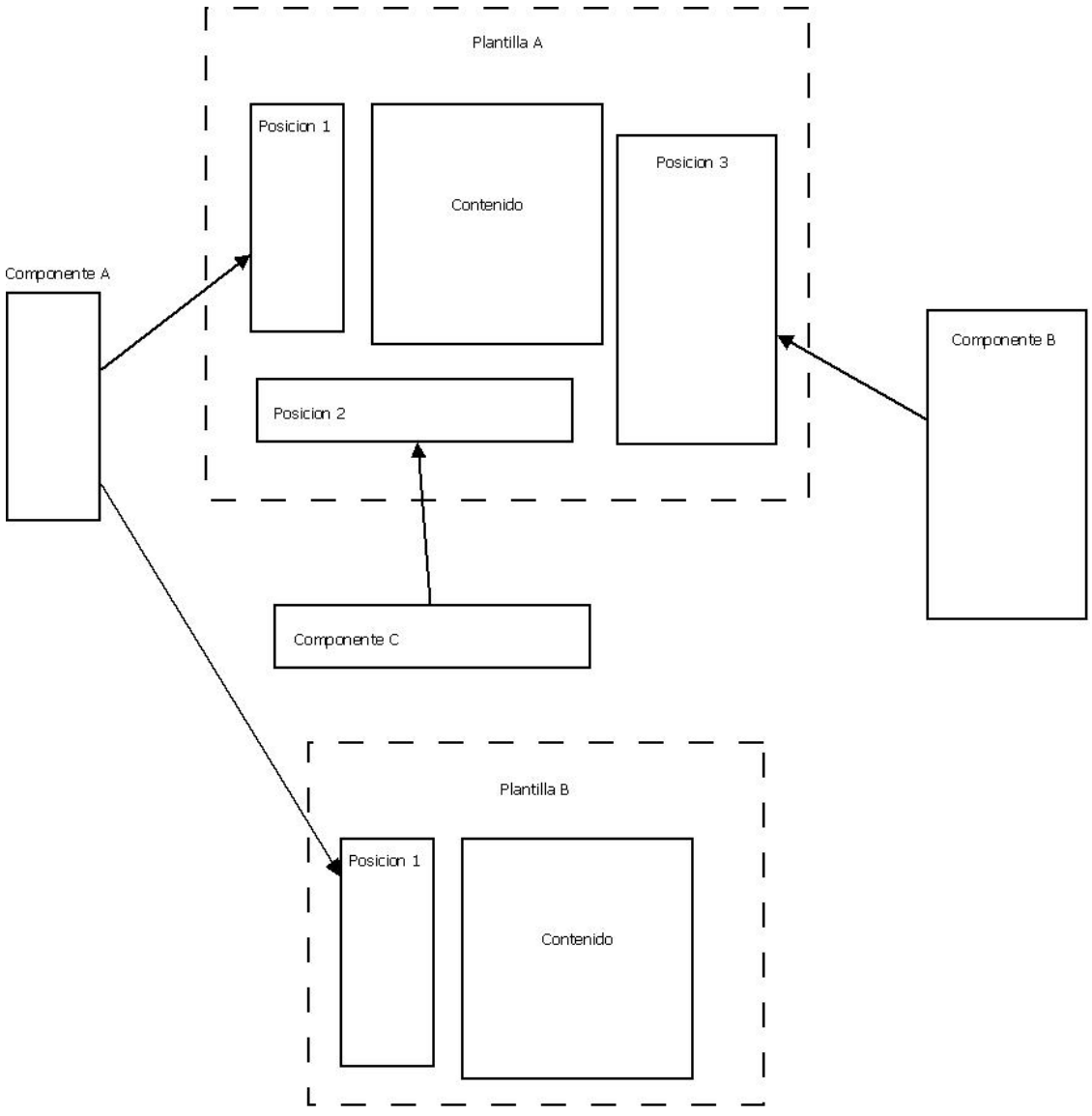
2 PROPUESTA TÉCNICA.

2.1 Visión Global.

Para cubrir las funcionalidades propuestas, se propone la siguiente solucion para el desarrollo del CMS:

- El CMS será un contendor de plantillas, de las cuales una siempre estará activa.
- Cada plantilla tendrá además de un espacio reservado para el contenido de la pagina que se este visualizando un numero determinado de posiciones donde se podrán incrustar diferentes componentes.
- Cada componente será una pagina con sus datos y su lógica propia y completa, este componente podrá influir sobre el contenido principal de la pagina y tratar datos.
- Estos componentes se visualizaran en las posiciones que el administrador haya determinado, podrán mostrarse en varias posiciones y en diferentes plantillas.

Un pequeño esquema del CMS puede representarse de la siguiente manera.



Desde el punto de vista de la lógica de ejecución el CMS cuenta con 3 Beans que controlan la

aplicación:

- **NavegacionController**: Este Bean se encarga de decidir que se va a mostrar en el navegador, su función es buscar que plantilla esta activa, que posiciones tiene esa plantilla, los componentes seleccionados para las diferentes posiciones y los destinos accesibles desde cada botón de la página.
- **SessionController**: Este Bean controla las propiedades y atributos activas en cada sesión de la aplicación, el atributo mas representativo es el usuario de la aplicación pero es accesible desde cualquier componente y puede almacenar y mostrar cualquier atributo que los diferentes componentes quieran manejar a nivel de sesión.
- **AplicationController**: Este Bean controla las propiedades y atributos activas a nivel de la aplicación, es accesible desde cualquier componente y puede almacenar y mostrar cualquier atributo que los diferentes componentes quieran manejar a nivel de sesión.

Además de estos Beans cada componente puede instalar su propio bean en el CMS que maneje la lógica del propio componente, o de varios componentes, estos bean pueden actuar a nivel de aplicación sesión o respuesta aunque es aconsejable que solo actúen a nivel de respuesta, almacenando los diferentes atributos en los beans controladores.

Por ultimo se han desarrollado varios componentes iniciales para el CMS, que realizan una serie de funciones básicas, orientadas a su administración y al testeo de funciones del CMS. Estos componentes son:

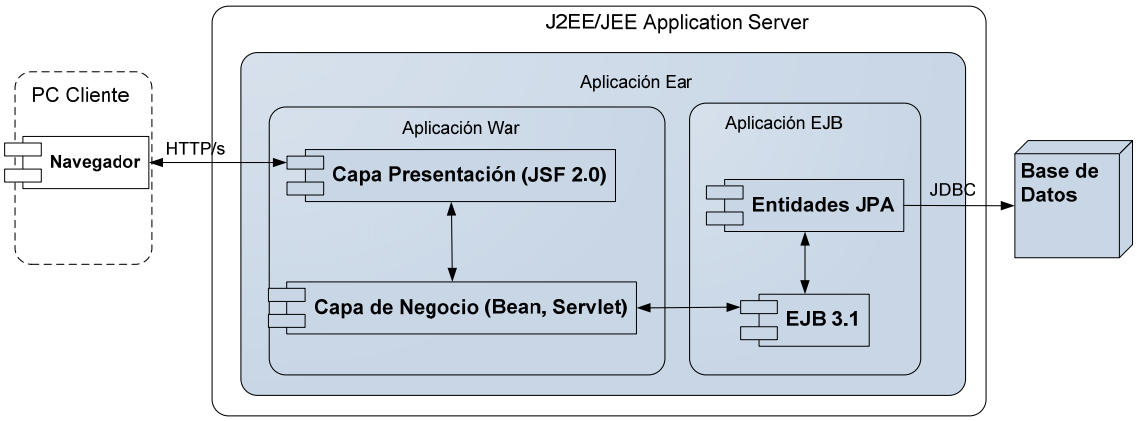
- Administración.
  - Gestor de tablas CMS.
  - Gestor de usuario y sesión.
  - Componente instalador de componentes.
- Componentes de Testeo:
  - Listado de categorías articulo.
  - Visualización de categorías.
  - Visualización de artículos.
  - Visualizador de imágenes.
  - Lector RSS.
  - Gestor de notas.
  - Visualizador de videos Youtube.

2.2 Diseño Lógico.

2.2.1 Descripción.

El diseño lógico que sigue el CMS, y en el que se deben basar todos los componentes consiste en una capa de presentación, generalmente JSF 2.0 o html, una capa de negocio ya sea un ManagedBean o un servlet, y la capa de acceso a datos, que la proporcionan un conjunto de EJB 3.1 que atacan a unas entidades JPA 2.0 relacionadas con la base de datos:

2.2.2 Esquema.



2.2.3 Componentes.

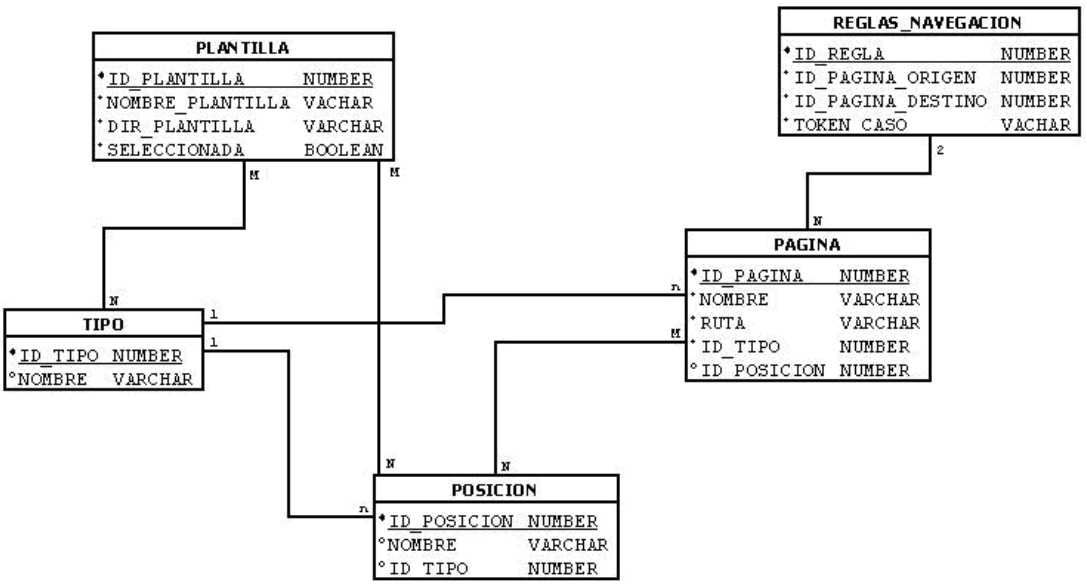
Los componentes serán siempre como parte de la aplicación WAR, es decir solo contendrán directamente al menos capa de presentación y de negocio, no pudiendo incluirse EJB o entidades JPA, aunque si podrán usar los ya existentes.

Para poder dotar a los componentes de acceso a la base de datos existe un EJB que permite realizar consultas JPQL y consultas SQL (también modificaciones, inserciones y borrados), sobre la base de datos, pudiendo así acceder a la capa de datos tanto a tablas y campos por defecto como a los creados durante la instalación de componente.

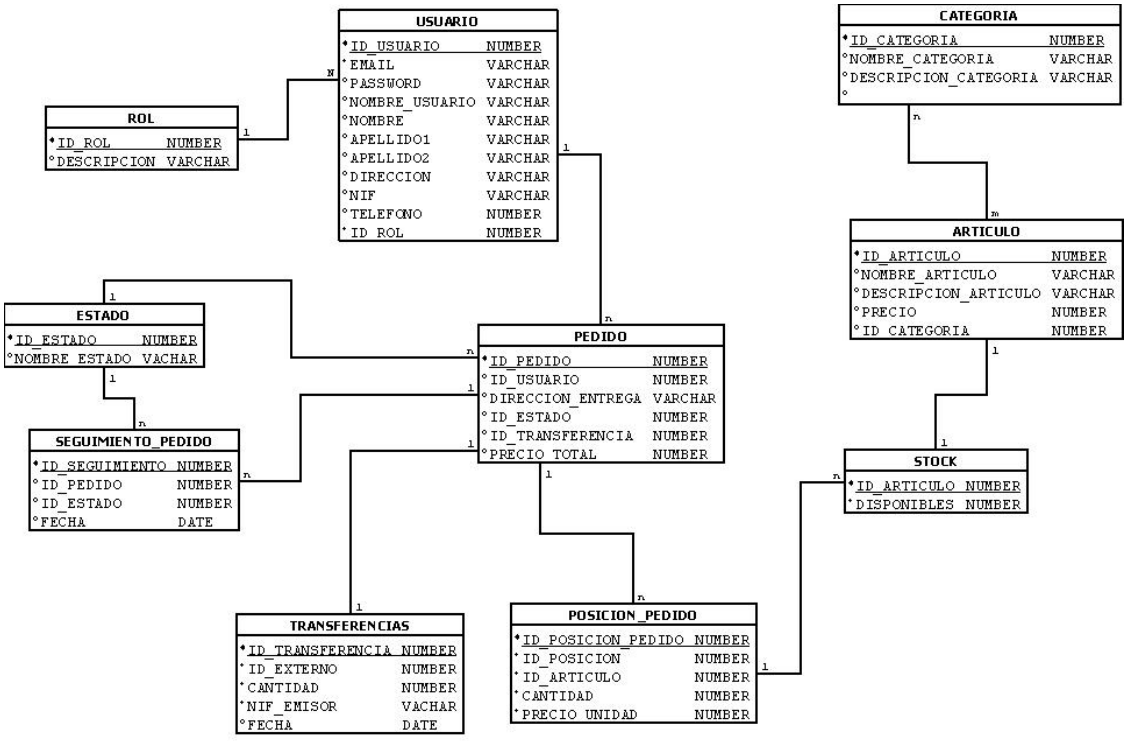
2.3 Modelo de datos.

El modelo de datos sigue se divide en dos espacios, esta el modelo de datos del CMS que gestiona los usuarios las plantillas y los componentes y el otro el soporte incluido en el CMS para desarrollar la tienda WEB, a continuación se detallan.

2.3.1 Modelo de datos para la gestión del CMS



2.3.2 Modelo de datos para el soporte a tienda Web



2.4 Funcionalidades CMS.

2.4.1 Selector dinámico de destinos.

2.4.1.1 Descripción Funcional.

Las diferentes páginas de contenido y los componentes poseen como es habitual un conjunto de Links a otras páginas, que establecen una navegación dentro de las paginas del sistema.

Estos Links pueden usar un destino fijo, incrustando la página de destino en el link o llamar al bean de navegación, indicando la acción que se ha realizado para que devuelva la respuesta fijada en el sistema a dicha acción.

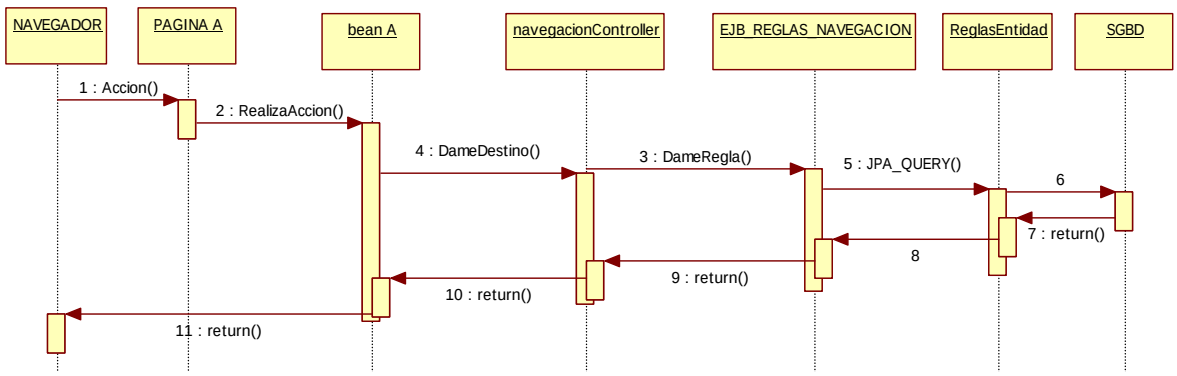
Esto es útil por ejemplo para diferentes paginas que contienen un link común por ejemplo registrar usuario, esta acción llevara a la pagina de registro de usuario y incrustar el método del sistema en vez de la pagina de destino, provocara que podamos configurar de una manera centralizada la pagina que nos mostrara para realizar esa acción.

También permitirá, alterar los flujos normales de las paginas, pudiendo meter paginas adicionales que realicen acciones adicionales dentro del flujo de un proceso complejos.

2.4.1.2 Análisis técnico.

Esta funcionalidad es un caso de uso especial del caso de uso general de la aplicación, al que llamaremos realizar acción y que incluye todas las interacciones con el sistema.

El flujo de esta funcionalidad es el siguiente:



Solo existe un flujo en los returns ya que aunque exista un error en el acceso al EJB o en la respuesta de la base de datos, se controla internamente en la clase que le llama, devolviendo la pagina de error por defecto en el return, en vez de la paginad dada por la base de datos, evitando así los posibles flujos del rollback.

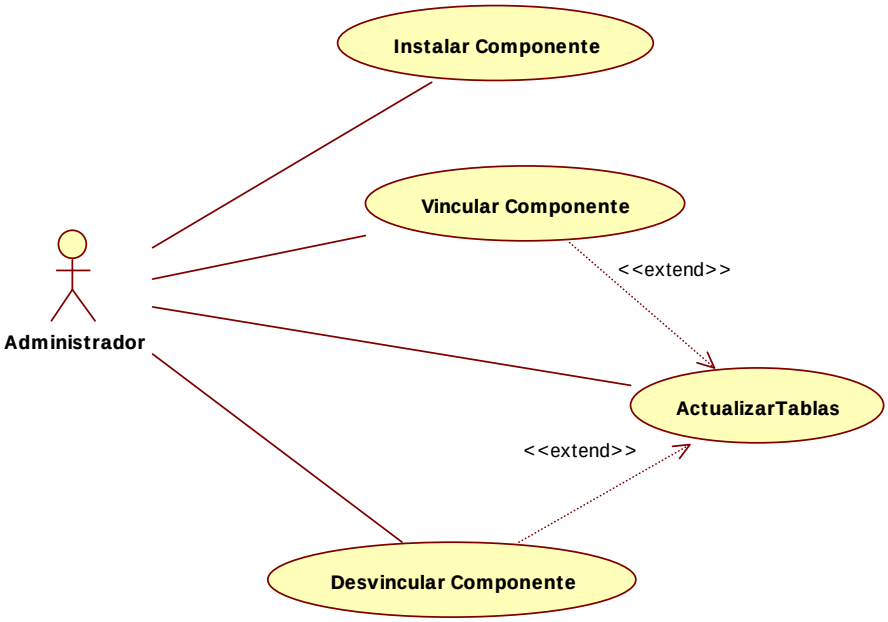
2.4.2 Agregar y quitar componentes.

2.4.2.1 Descripción Funcional.

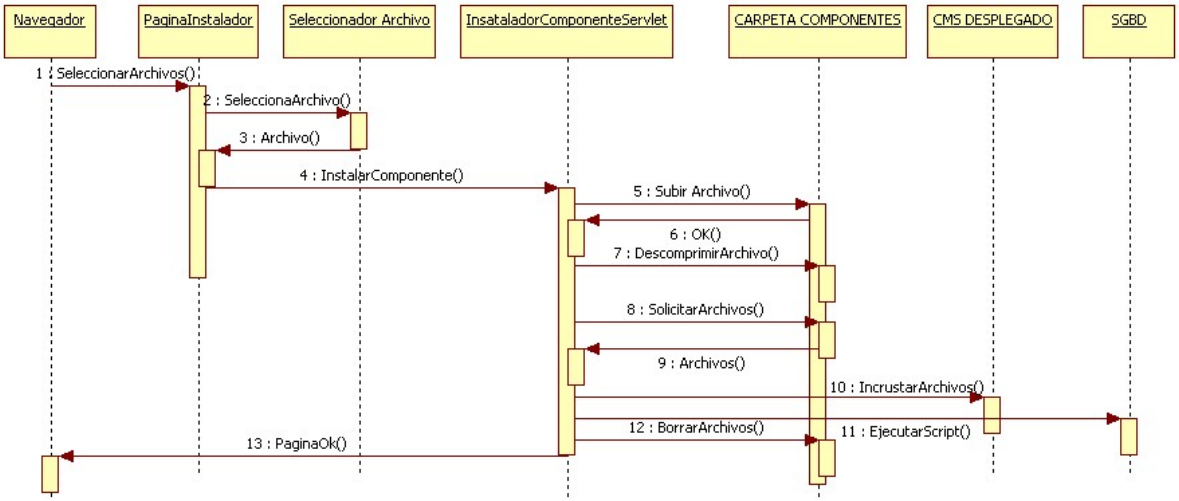
Esta es una de las funcionalidades propias de un CMS, poder instalar, agregar u ocultar contenidos. En este orden, el CMS es capaz de instalar diferentes componentes a través del interfaz Web del administrador y posteriormente a través de la gestión Web de las tablas del CMS, vincular estos componentes a diferentes posiciones de las plantillas.

2.4.2.2 Análisis técnico.

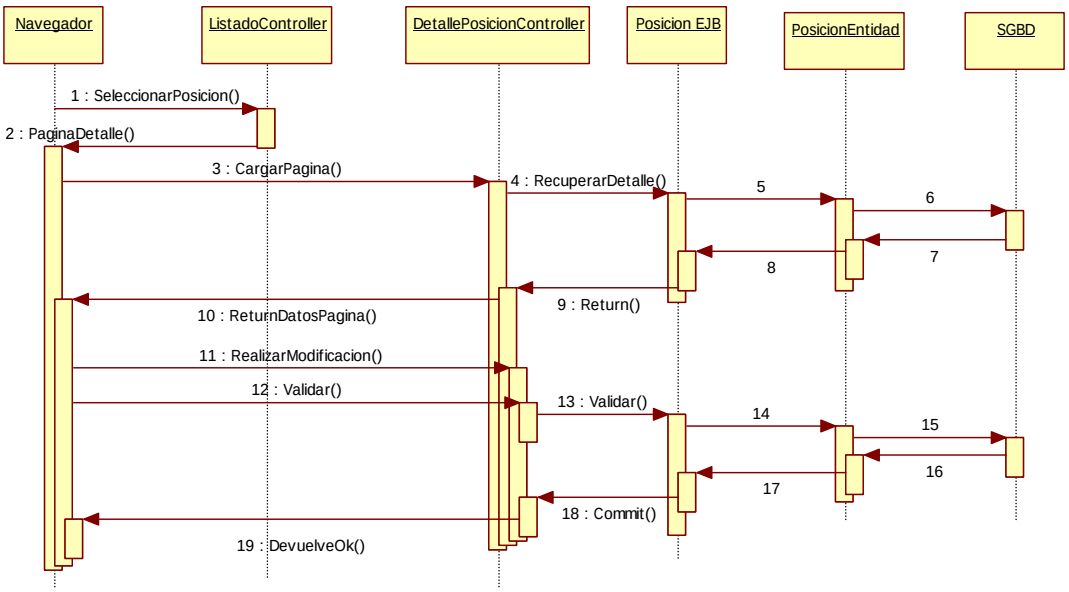
Esta funcionalidad como ya hemos visto se divide en diferentes casos de uso del sistema:



Instalar Componente.



Actualizar Tabla.



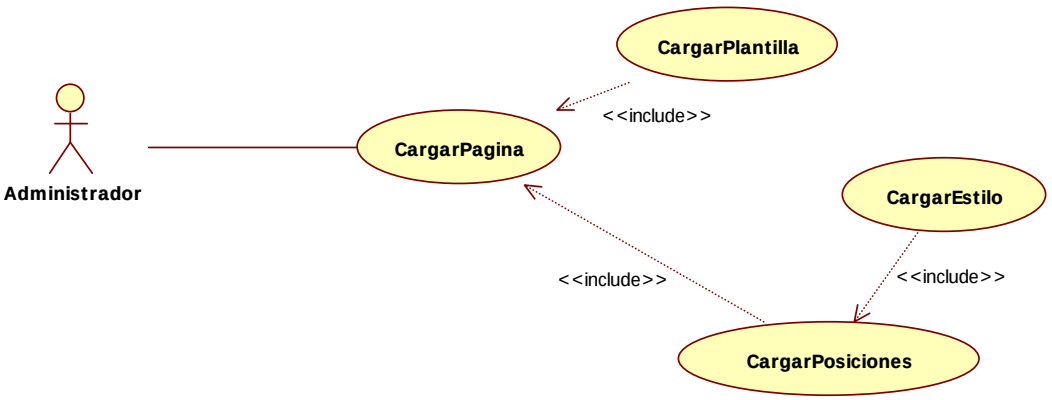
2.4.3 Cambio de plantillas del CMS.

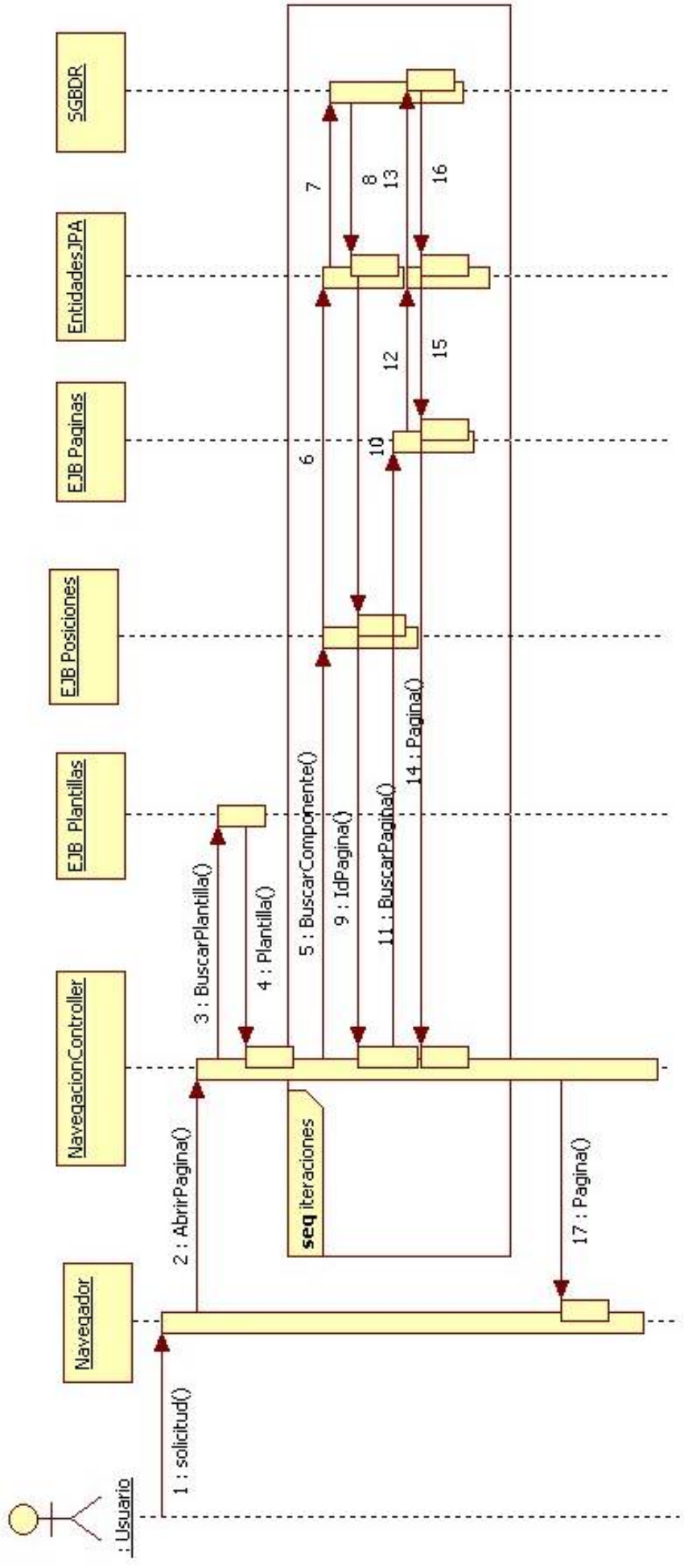
2.4.3.1 Descripción Funcional.

El CMS permite la selección dinámica de las plantillas en las que se basa la presentación final del Front end y el back end. Para ello cada página de contenido que se va a presentar tiene una cabecera en la que hace referencia a la plantilla con la que se debe cargar, esta plantilla se carga dinámicamente, reverenciando al navegador controller que busca en la base de datos cual es la plantilla seleccionada y la carga.

Para gestionar cual es la plantilla seleccionada podemos usar el gestor de tablas del modulo de administración, marcando la plantilla que queramos en la tabla de plantillas con el campo seleccionado.

2.4.3.2 Análisis técnico.





2.4.4 Cambio de estilos css.

2.4.4.1 Descripción Funcional.

El CMS permite la selección dinámica de estilos. En un principio el CMS admite una configuración de estilo para cada sesión, aunque los componentes desarrollados solo permiten la configuración de un estilo por aplicación. (Se puede desarrollar un componente que pueda configurar un estilo a nivel de sesión.

La implementación de esta funcionalidad consiste en que existe una entidad posición llamada estilo para cada entidad plantilla de nuestro CMS. Esta entidad representa el estilo activo en la plantilla y esta enlazada a una entidad pagina que es un archivo css con los estilos asociados a la plantilla.

Cada vez que se cargue una página, se buscara automáticamente la plantilla asociada una vez encontrada esta plantilla buscara automáticamente el estilo asociado. Estas búsquedas lo hará el JavaBean NavegacionController a través de los métodos getPlantilla y getEstilo.

Para modificar el estilo podremos usar la gestión de tablas para modificar la página asociada a la posición de la plantilla que representa el estilo.

2.4.4.2 Análisis técnico.

Para conseguir una carga dinámica de los estilos cada vez que se carga una página se realiza la búsqueda del estilo actual, siguiendo el proceso de búsqueda de plantilla, búsqueda de componentes asociados a la plantilla y entre los componentes se carga el estilo.

Los diagramas son los correspondientes al punto 2.4.3

2.4.5 Colocación de componentes en diferentes posiciones.

2.4.5.1 Descripción Funcional.

El CMS permite la selección de componentes para las diferentes posiciones de las plantillas que se pueden usar. Esto permite la configuración desde el modulo de administración de los componentes que se desea usar y visualizar en la presentación de las paginas web del Front-end.

La implementación de esta funcionalidad esta basada en la entidad posición, de la que se componen las plantillas, estas posiciones tienen un lugar asignado en la plantilla a través de un jsf:define, a su vez a estas posiciones se les asocia con una clave externa el componente que se desea visualizar, este componente tiene una pagina principal que es la que se muestra en la posición donde esta asignado.

Cada vez que se va a cargar una pagina la plantilla llama a el bean de navegación que recupera de la base de datos a través del EJB de plantillas que pagina (ruta) debe cargarse en cada posición asociada a la plantilla.

Para modificar que componente esta asociado a que posición, utilizaremos el gestor de tablas del back-end de administrador.

2.4.5.2 Análisis técnico.

Los diagramas son los correspondientes al punto 2.4.3

2.4.6 Sistema de instalación.

2.4.6.1 Descripción Funcional.

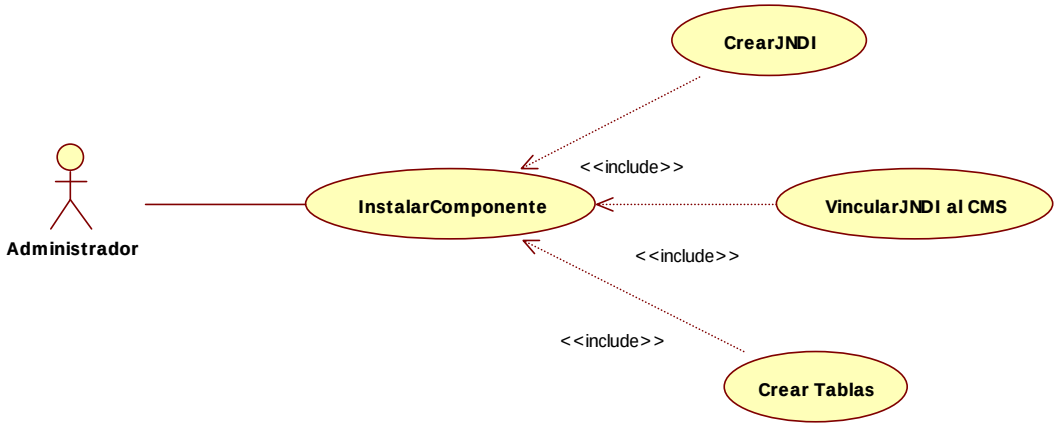
Se ha habilitado un sistema de instalación para el CMS, que permite su puesta en marcha y configuración sin tener que gestionar manualmente los XML con la configuración del CMS, ni la base de datos ni el servidor de aplicaciones.

Este CMS permite crear el pool de conexiones necesario para el funcionamiento del CMS, introducirlo en la configuración de la aplicación y crear en la base de datos asociada al pool un esquema con las tablas y una configuración por defecto para el CMS.

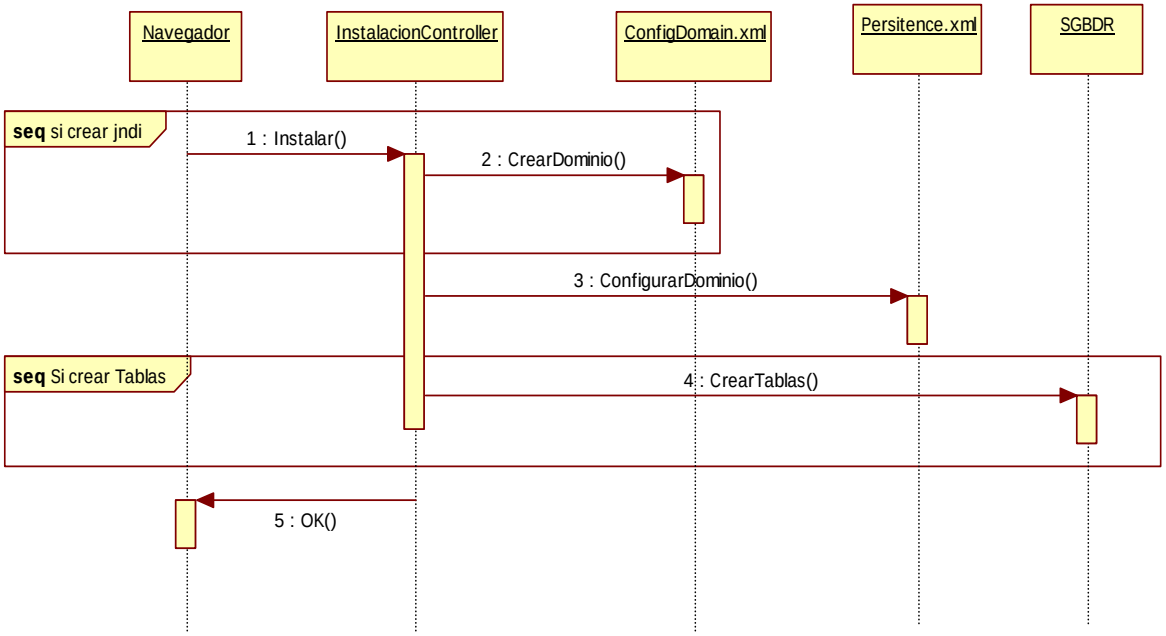
Su funcionamiento consiste en una pagina inicial en la que se rellenan los datos asociados a la conexión de la base de datos y el pool, se decide si se quiere crear nueva o usar alguna ya existente y un backbean, ejecuta con esos datos la creación del pool configurando el domain.xml y la creación de la base de datos a través de un script lanzado por JDBC.

La configuración del pool solo funciona para glassfish quedando como mejora futura la configuración en otros servidores de aplicaciones.

2.4.6.2 Análisis técnico.



La secuencia seria la siguiente:



2.4.7 Sistema de gestión de Usuarios.

2.4.7.1 Descripción Funcional.

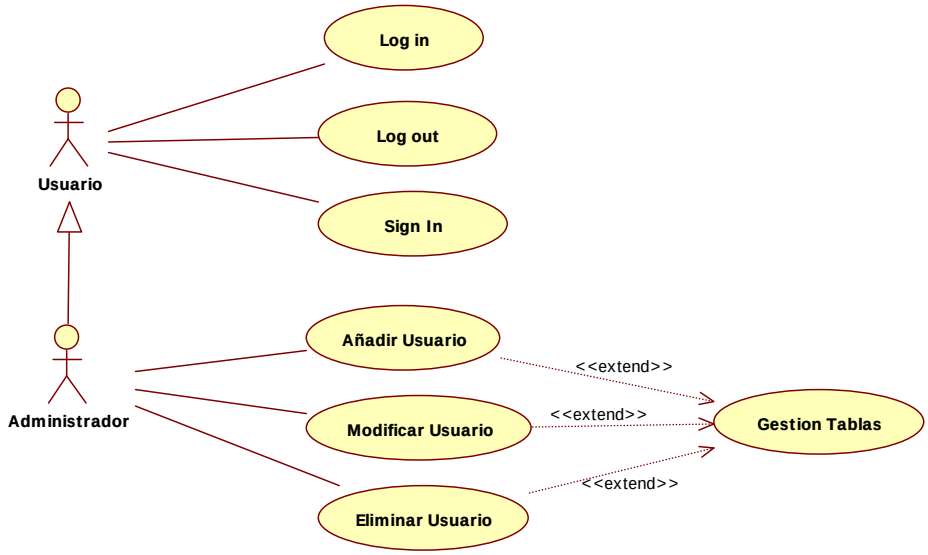
El sistema es capaz de mantener sesiones, registrar usuarios, cambiar sus roles, modificarlos y eliminarlos.

Para ello se ha implementado un componente por defecto capaz de iniciar una sesión en el sesioncontroller, registrar usuarios, y cerrar sesiones. Además el gestor de tablas puede cambiar los diferentes atributos de un usuario registrado, incluido el rol, además de crear y borrar usuarios.

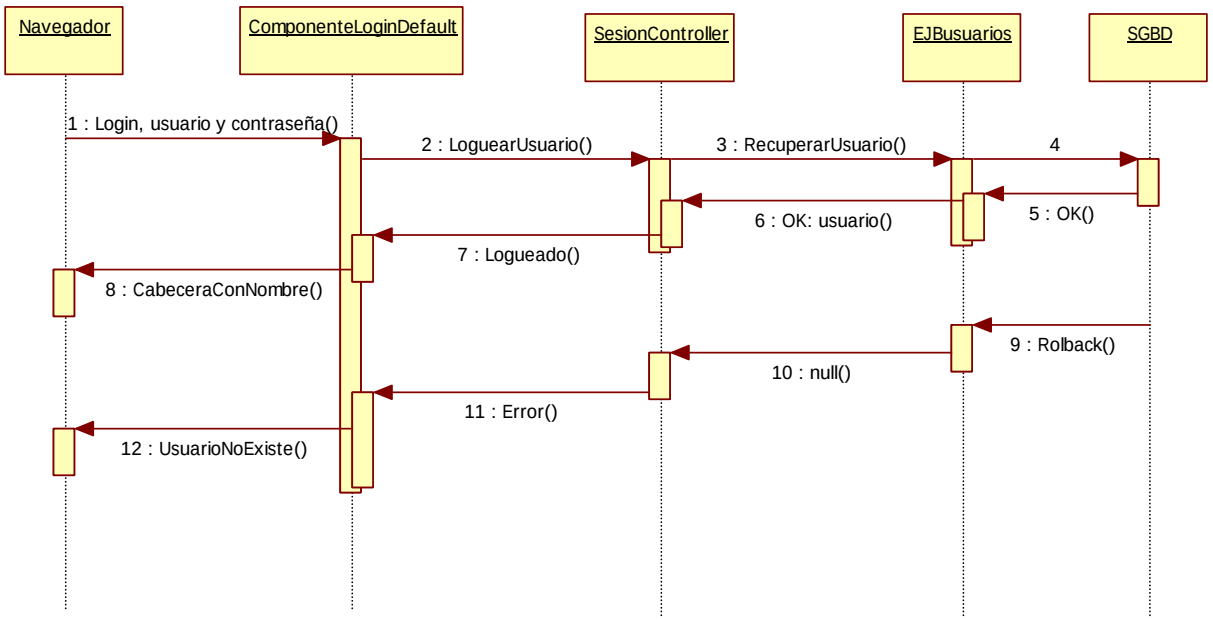
El componente que se ha implementado solicita el usuario y contraseña del usuario, con el llama al sesioncontroller, que busca un usuario valido y lo da de alta en la sesión, si no lo encuentra devuelve un error y el componente muestra el error en pantalla.

El componente también es capaz de registrar un nuevo usuario, solicitando los datos asociados al usuario que se va a registrar (menos el rol) y insertando en la base de datos a través del EJB de usuarios.

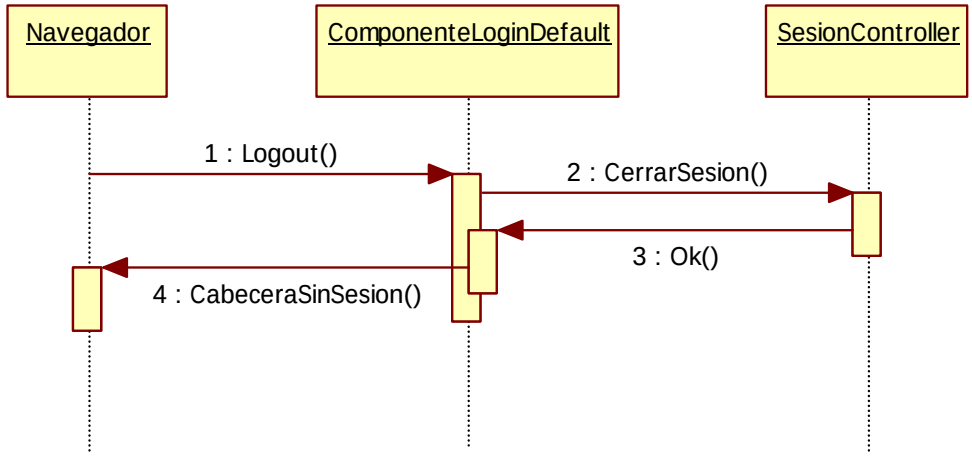
2.4.7.2 Análisis técnico.



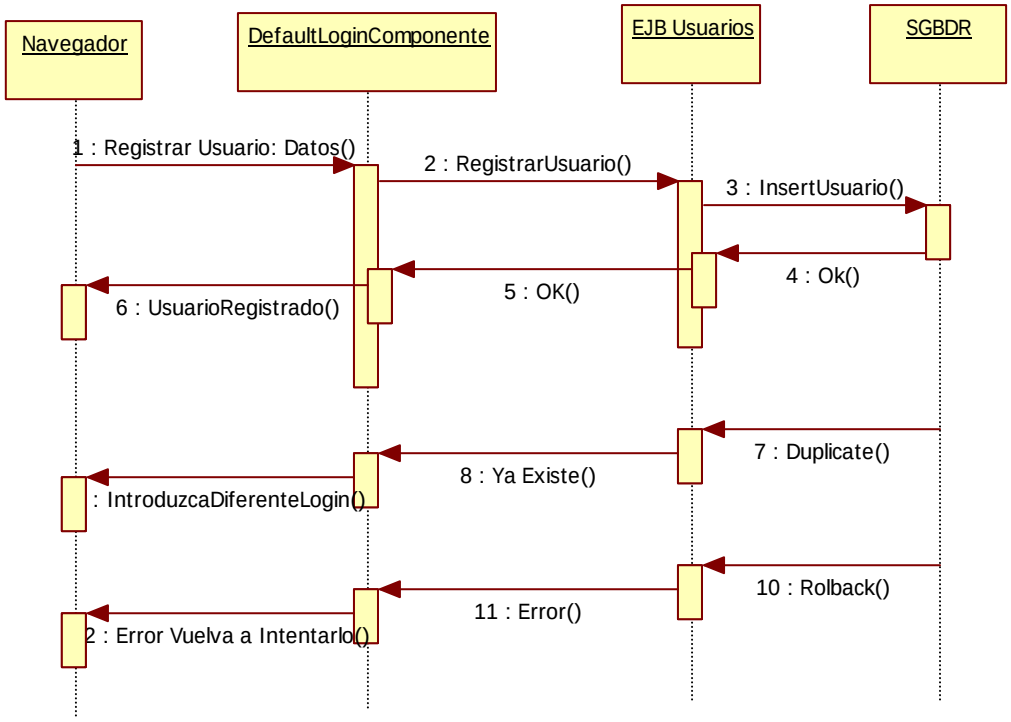
Secuencia LOGIN:



Secuencia LogOut:



Secuencia SignIn:



2.4.8 Inclusión de componentes básicos para tienda WEB.

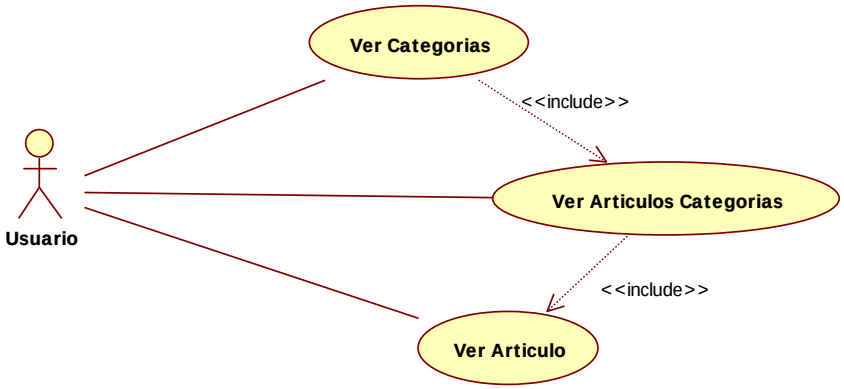
2.4.8.1 Descripción Funcional.

El CMS trae por defecto una serie de componentes para dar soporte a una tienda web, estos componentes montan una completa base de datos para la gestión de una tienda WEB y un visor por defecto para los artículos y sus categorías. Aunque la función de hacer pedidos no se ha implementado, podría enfocarse como un componente más.

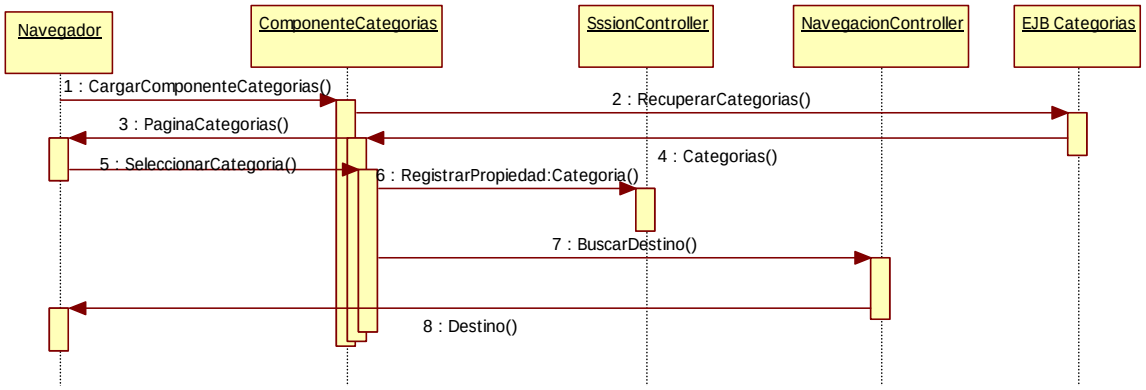
2.4.8.2 Análisis técnico.

El análisis técnico se basa en el análisis del componente que permite visualizar las categorías y sus artículos. Este componente hace uso del selector dinámico de destinos. Consiste en una página posicionable que enumera las categorías existentes y permite seleccionarlas para mostrar en la página de contenido una enumeración de los artículos de la categoría, que a su vez permite mostrar al seleccionar un artículo.

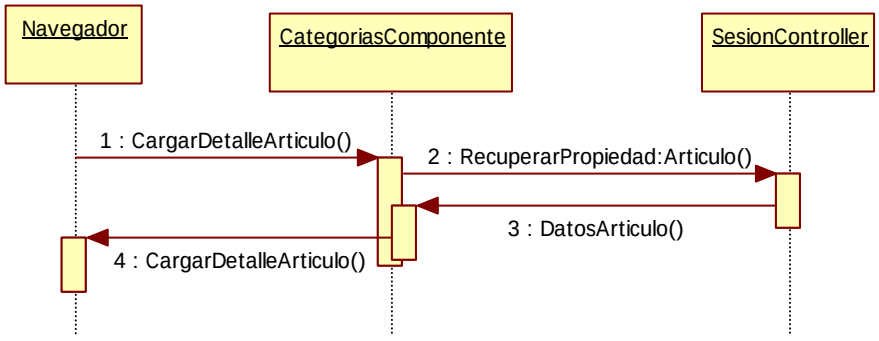
A continuación se detallan los casos de uso y su flujo por defecto (este flujo puede variar si se altera la navegación desde la selección dinámica de destinos.



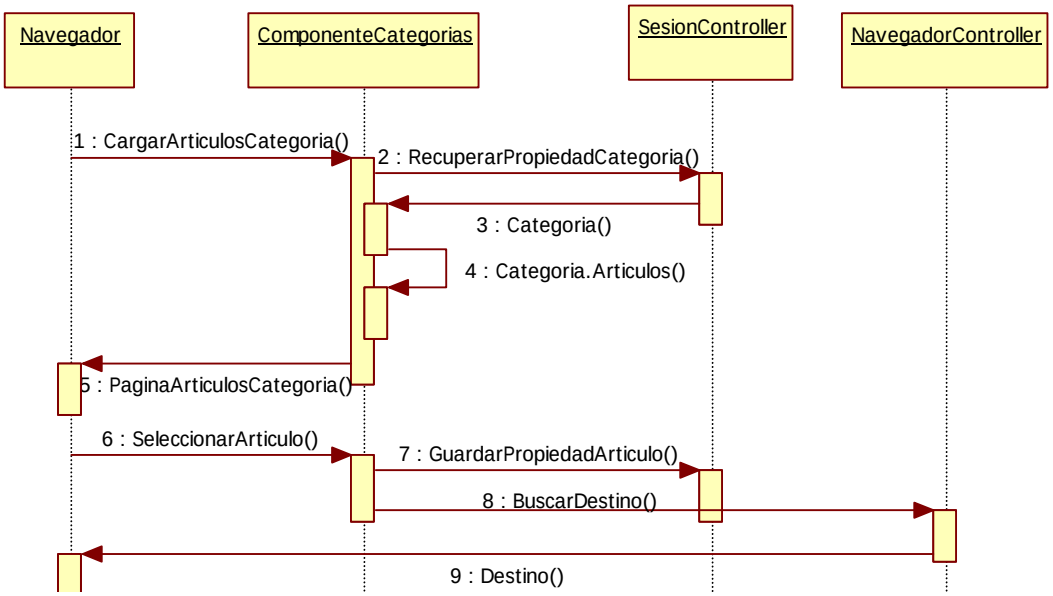
Secuencia Ver Categorías:



Secuencia DetalleArticulo:



Secuencia Ver Artículos Categorías:



2.4.9 Soporte a componentes basados en JSF 2.0, javascript, HTML, xhtml y servlets.

2.4.9.1 Descripción Funcional.

El CMS permite la instalación de HTML, servlets y CSS, scripts, beans… tal como se describe en la funcionalidad de instalación de componentes y como se detallara en la sección de componentes a continuación. Estos componentes se pueden ejecutar mas tarde debido a que se usan anotaciones en las clases en vez de configuración XML para su uso por lo que no existen dependencias externas de configuración para las clases. Además como el CMS se aloja en un servidor de aplicaciones se acepta el uso de beans además de servlets pues esta implementación está recogida en el servidor.

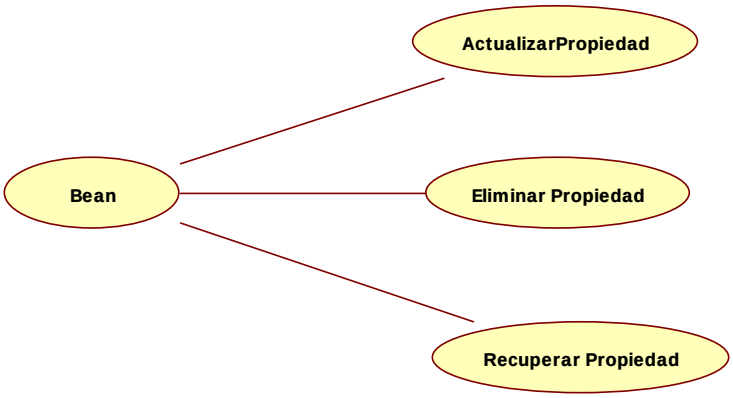
2.4.10 Guardar Propiedades a nivel de sesión // aplicación.

2.4.10.1 Descripción Funcional.

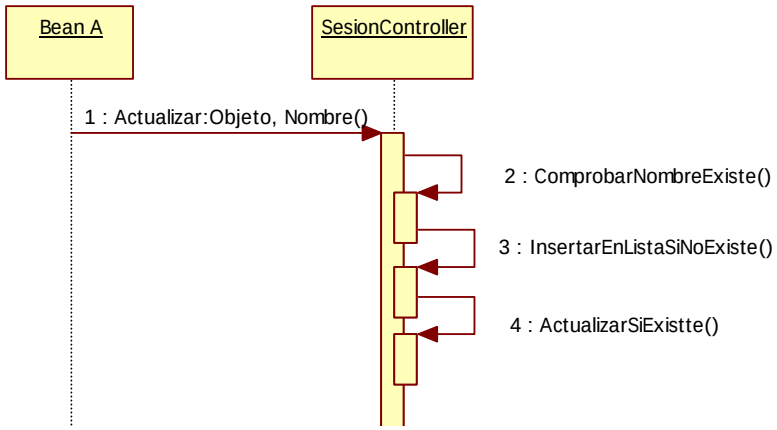
El Sistema permite que los componentes almacenen a nivel de sesión y a nivel de aplicación, datos para que puedan ser recuperados mas adelante o para ser recuperados por otros componentes. Esto permite compartir datos entre diferentes componentes y paginas y llevar unos registros de sesión mas completos.

2.4.10.2 Análisis técnico.

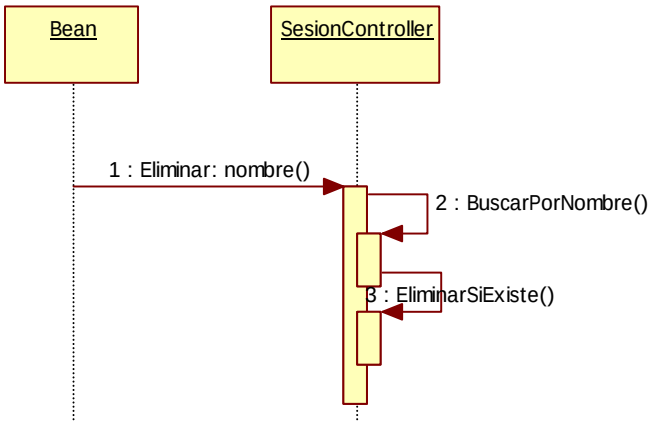
Para implementar esto existe un sesionController y un Application controller que son beans a nivel de sesión y aplicación respectivamente. Estos beans son accesibles desde otros beans y tienen una Lista de objetos con un nombre cada uno que puedes actualizar insertar modificar y eliminar con los métodos actualizar (actualiza o inserta), eliminar y recuperar (a través del nombre del objeto).



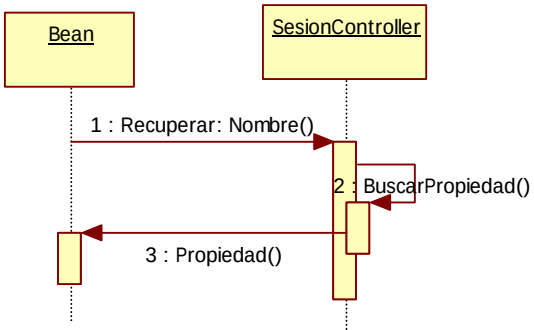
Secuencia Actualizar:



Secuencia Eliminar:



Secuencia Recuperar:

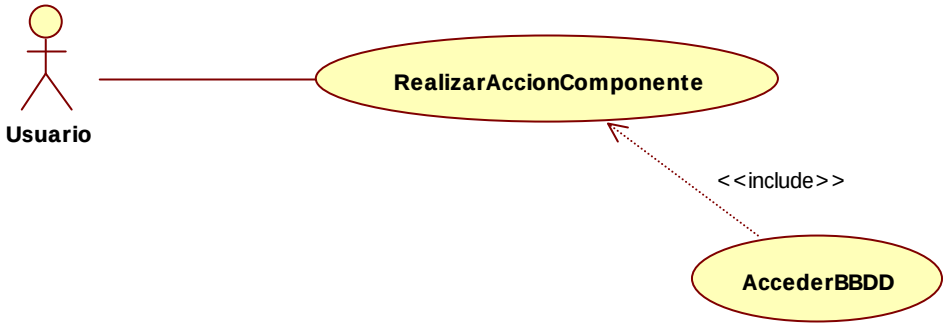


2.4.11 Interfaz de acceso a la base de datos para los componentes.

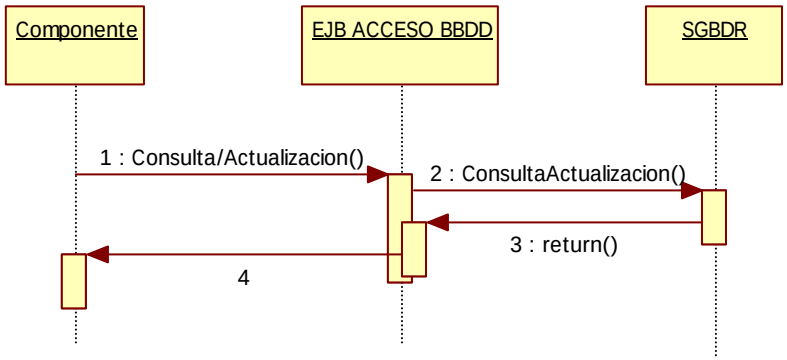
2.4.11.1 Descripción Funcional.

Dado que se permite ampliar la base de datos del CMS para dar soporte a los componentes instalables, se ha habilitado un EJB que permite hacer consultas y actualizaciones a la base de datos usando SQL JPQL. Esto permite abstraer la capa de datos de la parte WAR del proyecto. El EJB es necesario ya que el resto de los EJB solo permiten consultar y actualizar la tabla apoyada en la entidad asociada al EJB.

2.4.11.2 Análisis técnico.



Secuencia



2.5 Los componentes.

Los componentes son una serie de paginas HTML que completan la funcionalidad del CMS, existen varios tipos de componentes unos que añaden plantillas y estilos y otros que añaden paginas y reglas de negocio, estos componentes son instalables y seleccionables, dando una completa extensibilidad al proyecto.

2.5.1 Tipos.

2.5.1.1 Plantillas.

Este tipo de componentes consisten en una serie de plantillas seleccionables para el proyecto CMS, consisten en una plantilla HTML, un Script SQL que registra la pagina en el listado de páginas del CMS, y si es necesario uno o varios ManagedBean, que controlen propiedades especificas de la plantilla.

2.5.1.2 Estilos.

Este tipo de componentes consisten en una serie de estilos CSS seleccionables para el proyecto CMS, consisten en uno o varios CSS, un Script SQL que registra los CSS en el listado de paginas del CMS, y si es necesario uno o varios ManagedBean, que controlen propiedades especificas del CSS.

2.5.1.3 Componentes posicionales.

Este tipo de componentes son aquellos que más tarde se pueden ubicar en alguna de las posiciones de la plantilla. Consisten en un HTML principal que será el que se posicione, un managedBean o varios que controlen las reglas de negocio, otra serie de HTML navegables a través del espacio de contenido del CMS y un Script SQL que registre el componente y si es necesario modifique adecuadamente la base de datos.

2.5.1.4 Componentes funcionales.

Este tipo de componentes son aquellos que no se posicionan en la plantilla pero que son accesibles a través del menú de componentes del CMS (menú ubicado inicialmente en el index pero que es posible invocar desde cualquier componente). Consisten en uno o varios HTML, un managedBean o varios que controlen las reglas de negocio y un Script SQL que registre el componente y si es necesario modifique adecuadamente la base de datos.

2.5.2 Desarrollo.

La principal ventaja del CMS es que se pueden desarrollar componentes por parte de terceros usando las últimas tecnologías del mercado JAVA, el sistema de desarrollo de componentes, viene especificado en el documento:

PFC CMS JAVA-GUIA DE DESARROLLO DE COMPONENTES.doc

2.5.3 Instalación.

La instalación de componentes se realiza a través de un componente por defecto llamado “Instalador de componentes”, se pueden crear otros componentes que implementen esta función pero el componente por defecto es completamente operativo.

La descripción de su uso y por tanto la instalación de componentes viene descrita en el documento:

PFC CMS JAVA-GUIA DE ADMINISTRACION.doc

2.5.4 Ejemplos desarrollados.

Para el funcionamiento básico se han desarrollado varios componentes por defecto que pueden ser sustituidos pero teniendo en cuenta que desarrollan tareas básicas para la lógica del CMS. Estos componentes han sido denominados componentes de administración.

También para poder mostrar ejemplos prácticos de ciertas características del CMS se han desarrollado una serie de componentes a modo de testeo. Estos componentes se describen a continuación:

2.5.4.1 Administración.

Gestor de tablas CMS.

Este componente es un componente que solo puede ser usado por el administrador, consiste en un gestor de las tablas por defecto del CMS, es un componente posicionadle en el que su HTML principal es un listado de las tablas por defecto del CMS, cuando pulsas sobre un elemento de la lista accede a un listado de los registro de esa tabla, en el espacio de contenido, desde este listado puedes crear y borrar registros y acceder al detalle de un registro en modo modificación o modo lectura.

Gestor de usuario y sesión.

Este componente posicionable permite el logueo y registro de usuarios, así como el logout. Su componente principal es un HTML que indica si el usuario esta logueado, y accesos a las opciones de login, logout and singin del usuario, estas opciones se realiza en unas páginas que el componente carga en el frame de contenido.

Componente instalador de componentes.

Este componente es un componente funcional, al que solo pueden acceder los administradores, permite cargar nuevos componentes en el CMS, consiste en un HTML, que deja seleccionar un archivo en local y lo carga al servidor, distribuyendo sus archivos y ejecutando los scripts contenidos en el cuando se pulsa el botón enviar.

2.5.4.2 Componentes de Testeo:

Gestión de categorías articulo.

Este componente es un componente posicionable, consiste en una pagina principal que lista las categorías existentes en la base de datos, y que al pulsar en un elemento, genera una pagina con la lista de artículos existentes en la base de datos, correspondientes a esa categoría, estos artículos a la vez son seleccionables, accediendo al detalle de los artículos en el frame de contenido. Todas estas relaciones de paginas se realizan a través de las reglas de navegación por lo que son gestionables.

Visualizador de imágenes.

Este componente posicionable, genera una pantalla en la posición seleccionable donde se puede ver una imagen estática.

Lector RSS.

Este componente posicionable, genera en la posición seleccionada, un lector de RSS de la pagina RegEdit, conocido gestor de noticias.

Gestor de notas.

Este componente funcional, permite generar notas en el CMS, y guardarlas en una tabla de la BBDD que crea durante su instalación, permite añadir notas, y después visualizarlas o modificarlas, consiste en un HTML, que lista las notas actuales, para poder acceder a ellas y permite también crear notas nuevas y guardarlas.

Visualizador de videos Youtube.

Este componente posicionable genera en la posición seleccionada, un visualizador de videos de youtube.

3 VALIDACIÓN Y CONCLUSIONES.

3.1 Validación Funcional.

Para validar el funcionamiento de las diferentes operaciones del CMS, se ha realizado una batería de pruebas que cubre todas las funcionalidades y que viene especificada en el documento:

PFC CMS JAVA- Validación Funcional.doc

3.2 Validación de Infraestructura.

Se han realizado 3 diferentes tipos de pruebas usando la herramienta Webserver Stress Tool que nos facilita realizar peticiones con diferentes usuarios simultáneamente y la medición de los tiempos.

La primera ha consistido en aumentar el volumen de la base de datos y ejecutar la aplicación ir navegando entre las diferentes páginas, estos son los resultados obtenidos: en 100 y en 1000 veces.

Estos son los diferentes resultados atenidos

registros:	10-100	100-1000	1000-10000
Segundos para carga de pagina gestión artículos	3.03	3.02	3.1

Con la carga masiva de la base de datos, se ha iniciado la instalación masiva de componentes y se ha vuelto a medir los tiempos de navegación, para ello se han duplicado componentes renombrando de diferentes maneras los que ya tenemos.

Componentes	3	10	20
Segundos para carga de pagina gestión artículos	3.01	3.01	3.01

Por ultimo con la carga máxima de la base de datos y 20 componentes instalados, se han realizado, peticiones cada 3 segundos a la página de datos obteniendo los siguientes resultados:

Peticiones	10	100	500
Segundos para carga de pagina gestión artículos	3.1	4	4.5

3.3 Conclusiones.

Una vez el proyecto ha sido terminado, somos capaces de afirmar, que el proyecto del CMS esta preparado para empezar a usarse, con los componentes de administración un usuario es capaz de gestionar completamente el CMS, los componentes de Testeo dan una muestra de todas las características que el CMS puede llegar a proporcionar a los componentes desarrollados y que disponemos de una tecnología java permite un desarrollo rápido y consistente. También hacer hincapié en que a nivel de carga del servidor, presenta una carga razonable para su uso.

Sin embargo la fuerza del CMS y su uso práctico dependen de la cantidad de componentes que se desarrollen para el CMS, ya que al ser un gestor de contenidos gran parte de su valor reside en los contenidos que va a gestionar.

3.4 Visión de futuro.

En este apartado vamos a describir que posibilidades se pueden plantear a este CMS para completar su funcionalidad.

Instalación de componentes en caliente.

Muchos de los componentes cuando se instalan necesitan de un reinicio para su funcionamiento, poder instalar componentes en caliente, aliviaría mucho las paradas de servicio y acercaría el CMS a un gestor compatible con la alta disponibilidad.

Gestor dinámico de tablas.

Este CMS mantiene una gestión dinámica de todos sus componentes y datos sin embargo cuando se crean nuevas tablas o campos en las tablas a través de los Scripts, el programa no añade automáticamente la gestión de estos nuevos campos o tablas, al gestor de tablas, debiendo manejarse desde el propio componente o desde la base de datos.

Estandarización del sistema de instalación.

Aunque el CMS es compatible con cualquier servidor de aplicaciones, su instalador, solo esta configurado para Glassfish 3.1, estandarizar este instalador permitirá instalar este CMS en cualquier servidor de aplicaciones, sin tocar ninguna configuración XML de manera manual.

Creación de una gama amplia de Componentes.

Este CMS trae consigo una lista de componentes básicos, sin embargo la potencia y flexibilidad que le caracterizan le permite soportar componentes mas complejos y completos que puedan servir al usuario final, tener una buena base de componentes, es clave para el CMS ya que por muy buen y potente administrador de contenidos que pueda ser, sin unos componentes completos y de calidad no aporta realmente nada a un usuario final.

ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN

Titulación:

INGENIERO EN INFORMÁTICA

Título del proyecto:

CMS-JAVA

Validación Funcional

Miguel Larraz Sala

Diego Pérez

Pamplona, Fecha de defensa

Control de cambios

Fecha	Autor	Descripción	Versión
22/12/2010	Miguel Larraz	Versión Inicial	0.1

Revisiones

Autor	Versión	Responsabilidad	Fecha

Índice de contenidos

.

1 INTRODUCCIÓN.

1.1 Objetivo del documento.

El objetivo de este documento es detallar la batería de pruebas que se ha realizado para validar el funcionamiento del CMS.

La realización de esta batería de pruebas asegura que el CMS funciona correctamente y con un comportamiento acorde a su diseño.

1.2 Sistema de pruebas.

Las pruebas funcionales se han diseñado en función de las acciones posibles del CMS y las respuestas esperadas a dichas acciones. El criterio que marca la superación de la prueba es que dada una acción del CMS se produzca la respuesta especificada en el documento de pruebas.

Las pruebas se han dividido en bloques según la funcionalidad general con la que la acción esta relacionada.

2 BLOQUES DE PRUEBAS.

2.1 Selección dinámica del flujo entre páginas.

2.1.1 Prueba 1: Navegación.

La primera prueba ha consistido en la navegación por el componente de listado de categorías, desde el listado de categorías, al listado de artículos hasta el detalle de artículo.

Este componente hace uso de las características de navegación dinámica, y la navegación correcta entre estas páginas indica que el controlador de navegación funciona correctamente.

2.1.2 Prueba 2: Cambio de Flujo.

En esta prueba hemos alterado el flujo de pagina del botón de cerrar sesión al botón de instalar componente, este cambio de flujo realizado en el gestor de tablas de la consola de administrador, provoca que al intentar cerrar sesión nos lleve a la pagina de cerrar componente, el resultado ha sido el esperado así que podemos darlo por validado.

Para completar esta prueba hemos cambiado el flujo de página a un destino que no tenemos actualmente funcionando, como resultado la aplicación nos ha llevado a la página de error, indicada por defecto en el CMS:

2.1.3 Prueba 3: Navegación después del cambio de flujo.

Después del cambio de flujo, hemos seguido navegando por el CMS en modo administrador, y volviendo a pulsar el LINK que hemos cambiado, llegando de nuevo a la página de instalador componente tal como se esperaba.

2.2 Agregar y quitar componentes.

2.2.1 Instalación de un componente.

Con el CMS recién instalado hemos instalado el componente de video de gestor de notas.

Tanto el HTML, el bean se han subido correctamente y el script de SQL se ha ejecutado quedando registrado el nuevo componente y añadida la tabla de notas a la base de datos.

2.2.2 Agregar componente a posición.

Hemos instalado el componente de lector de RSS y posicionado con el gestor de tablas en la parte derecha de la página.

Al abrir una nueva pestaña, el componente se muestra correctamente.

2.2.3 Eliminar Componente de posición.

Hemos quitado de la posición

2.3 Cambio de estilos CSS.

2.3.1 Instalación de un estilo.

Con el CMS recién instalado hemos instalado el estilo 2.

Tanto el HTML se ha subido correctamente y el script de SQL se ha ejecutado quedando registrado el nuevo estilo.

2.3.2 Cambio de Estilo.

Hemos posicionado el nuevo estilo en la posición de estilo de la plantilla usada y al abrir una nueva sesión el estilo de la página se ha ajustado a las nuevas características indicadas en el CSS.

2.4 Cambio de plantillas del CMS.

2.4.1 Instalación de plantilla.

Con el CMS recién instalado hemos instalado una nueva plantilla.

Tanto el HTML se ha subido correctamente y el script de SQL se ha ejecutado quedando registrado la nueva plantilla y sus posiciones.

2.4.2 Cambio de plantilla.

Hemos colocado componentes en las posiciones de la nueva plantilla y posteriormente la hemos seleccionado como plantilla por defecto.

Al abrir la nueva sesión, se ha mostrado la página siguiendo las indicaciones de la nueva plantilla.

2.5 Colocación de componentes en diferentes posiciones.

2.5.1 Agregar un componente a dos posiciones.

Hemos agregado el mismo componente a dos posiciones diferentes (izquierda y derecha), el componente (lector RSS) se ha mostrado sin problemas.

2.6 Sistema de instalación.

2.6.1 Proceder a instalación con nuevo JNDI.

Hemos realizado la instalación de la aplicación usando un nuevo conector a una base de datos ya existente con tablas existentes y la aplicación se ha instalado y empezado a usar sin problemas.

2.6.2 Proceder a instalación con JNDI existente.

Hemos realizado la instalación de la aplicación usando un conector JNDI existente a una base de datos ya existente con tablas existentes y la aplicación se ha instalado y empezado a usar sin problemas.

2.6.3 Proceder a instalación con creación de tablas.

Hemos realizado la instalación de la aplicación usando un nuevo conector a una base de datos ya existente sin esquema ni tablas y la aplicación se ha instalado y creado la base de datos sin problemas.

2.7 Sistema de gestión de Usuarios.

2.7.1 Agregar usuario.

Hemos agregado un usuario desde el apartado de registros y lo hemos empezado a usar, con completa funcionalidad.

2.7.2 Modificar atributo de usuario.

Hemos modificado el nombre de usuario desde la gestión de tablas y el proceso se ha realizado sin problemas.

2.7.3 Eliminar usuario.

Hemos eliminado el usuario desde la gestión de tablas sin problemas.

2.7.4 Agregar administrador.

Hemos dado categoría administrador a un usuario y el sistema lo ha reconocido correctamente y le ha dado la posibilidad de acceder a las zonas restringidas.

2.8 Soporte a componentes.

2.8.1 Agregar componente con JSF 2.0.

Hemos desarrollado componentes que usan JSF 2.0 y se han integrado sin problemas con el CMS (visualizador de imágenes).

2.8.2 Agregar componente con java script

Hemos desarrollado componentes que usan java script y se han integrado sin problemas con el CMS (Lector RSS).

2.8.3 Agregar componente con HTML

Hemos desarrollado componentes que usan HTML y se han integrado sin problemas con el CMS (youtube).

2.8.4 Agregar componente con, xhtml.

Hemos desarrollado componentes que usan xhtml y se han integrado sin problemas con el CMS (gestor de notas).

2.8.5 Agregar componente con servlets.

Hemos desarrollado componentes que usan servlets y se han integrado sin problemas con el CMS (instalador de componentes).

2.9 Interfaz de acceso a la base de datos para los componentes.

2.9.1 Acceso a listado de tablas.

Logueandonos con el administrador, hemos accedido al listado de tablas sin problemas.

2.9.2 Acceso a listado de registros.

Desde este listado hemos seleccionado una tabla y accedido a los registros que contiene.

2.9.3 Uso de paginación.

Hemos navegado entre las páginas de registro y comprobado que mantienen la integridad de los datos.

2.9.4 Alta de registro.

Hemos creado un nuevo registro en la tabla sin problemas.

2.9.5 Modificación de registro.

Hemos modificado un nuevo registro en la tabla sin problemas.

2.9.6 Visualización de registro.

Hemos entrado en un registro en la tabla sin problemas en modo visualización.

ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN

Titulación:

INGENIERO EN INFORMÁTICA

Título del proyecto:

CMS-JAVA

GUIA DE ADMINISTRACIÓN

Miguel Larraz Sala

Diego Pérez

Pamplona, Fecha de defensa

Control de cambios

Fecha	Autor	Descripción	Versión
22/12/2010	Miguel Larraz	Versión inicial	0.1

Revisiones

Autor	Versión	Responsabilidad	Fecha

Índice de contenidos

1 INTRODUCCIÓN.

1.1 Objetivo.

Este documento tiene como objetivo, detallar los diferentes procedimientos para realizar las operaciones de administración de la aplicación CMS desarrollada en el proyecto.

1.2 A quien va dirigido.

El documento esta dirigido a las personas que van a desarrollar el rol de administradores del CMS y que poseen por tanto un usuario administrador, con el que deberán estar logueados, para realizar estas operaciones.

1.3 Requisitos previos.

El CMS esta desarrollado para ejecutarse en un servidor de aplicaciones glassfish-3.0.1, además necesita interactuar con una base de datos de sintaxis similar a MySQL, estos 2 servidores (el de aplicaciones y el SGBD) deben estar instalados en el entorno donde se va a desplegar el CMS, ya sea en un terminar local, o en una red.

Los pasos básicos para instalar glassfish y MySQL se explican en los anexos al final del documento.

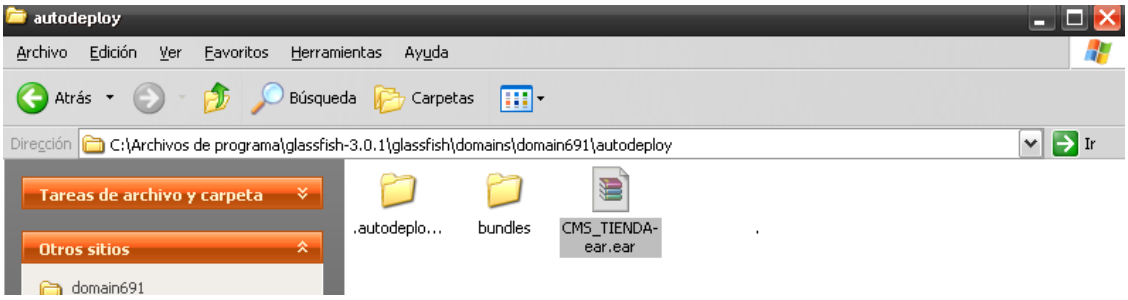
2 DESPLIEGUE Y CONFIGURACION DEL CMS.

- Para desplegar el CMS debemos dirigirnos a la carpeta donde este instalado glassfish a partir de ahora **Glassfish_home**.

- Una vez en esta carpeta debemos dirigirnos al path:
  - glassfish\domains**<DOMAIN>**\autodeploy

  <DOMAIN> se refiere al nombre del dominio donde queramos instalar el CMS.

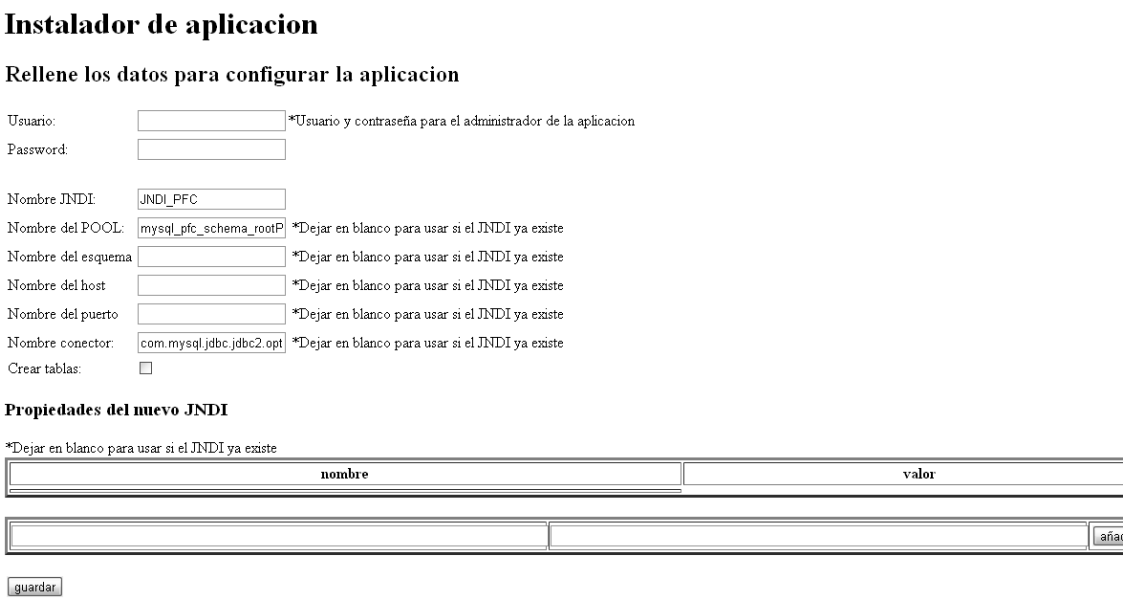
- En esta carpeta deberemos dejar el desplegable **CMS_TIENDA_EAR.EAR**



- En este punto, el servidor de aplicaciones desplegará la aplicación, y deberemos acceder a la pantalla de configuración, para ello accedemos a la ruta:

http://<maquina>:<puerto>/CMS_TIENDA_INSTALADOR

- Donde se nos cargará la siguiente página:



- En esta página configuraremos la conexión a la base de datos. Para ello rellenaremos:
  - **Usuario**: Usuario de la base de datos.
  - **Password**: Password de la base de datos.
  - **Nombre JNDI**: Nombre del JNDI de glassfish asociado a la base de datos.
  - **Nombre POOL**: Nombre del POOL de glassfish en el caso que se deba crear uno nuevo.
  - **Nombre esquema**: Nombre del esquema de la base de datos en el caso de que se deba crear uno nuevo.
  - **Nombre del host**: Nombre del host donde esta alojada la base de datos.
  - **Puerto**: puerto donde esta a la escucha de la base de datos.
  - **Nombre conector** Nombre del conector con la base de datos (deberá estar instalado en las librerías de glassfish en el caso de no ser MySQL.
  - **Crear Tablas**: Se marcara si se quiere que la aplicación cree automáticamente las tablas de la aplicación en la base de datos.
  - **Propiedades adicionales**: Propiedades adicionales para el pool de glassfish (timeout…).

Si no se desea crear un nuevo JNDI sino usar uno ya existente, no se deberá rellenar ningún campo relacionado con el POOL, ni la dirección de la base de datos.

- Finalmente pulsaremos guardar y reiniciaremos el servidor cuando aparezca la página de confirmación.
- Después de que el servidor sea reiniciado se podrá acceder a la aplicación en la ruta:

http://<maquina>:<puerto>/EjemploPlantilla/

3 INSTALACIÓN DE COMPONENTES.

Para realizar el proceso de instalación de un componente necesitaremos tener un usuario con el rol administrador. Con este usuario podremos cargar los nuevos componentes para el CMS desde la ventana habilitada para ello, siguiendo estos pasos:

- Deberemos loguearnos en la aplicación, para ello y suponiendo que este activo el componente por defecto para la gestión de usuarios:
  - Pulsaremos sobre el botón “Iniciar conexión”, situado en la parte superior izquierda:

  - Rellenaríamos nuestro usuario y contraseña y pulsaríamos acceder:

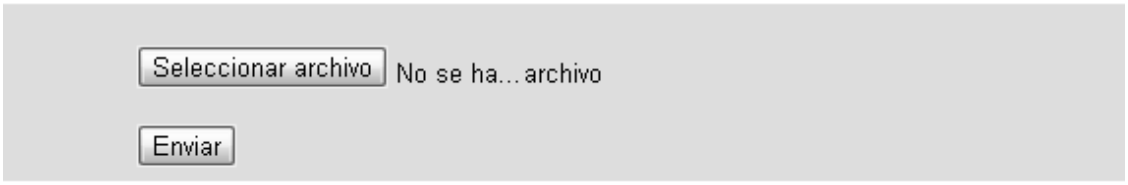
Por defecto: Usuario: Admin y Password: Admin

- Para acceder a la pagina de instalación, podríamos ir desde el menú principal pulsando en Bienvenido y después en la opción instalar componentes, o directamente por la ruta:

<APLICACIÓN>/ faces/InstaladorComponente.xhtml

- Para seleccionar el componente a instalar pulsaríamos “Seleccionar archivo” .

- Por ultimo presionaremos “Enviar”.



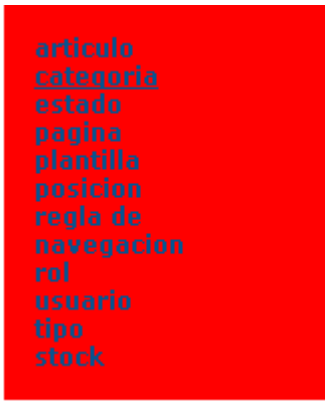
4 SELECCIÓN DE PLANTILLA.

Las plantillas en las que se basa el CMS son instalables y seleccionables, el proceso de instalación es el mismo que el de cualquier otro componente y para su selección se deben seguir los siguientes pasos:

- Deberemos loguearnos en la aplicación, para ello y suponiendo que este activo el componente por defecto para la gestión de usuarios:
  - Pulsaremos sobre el botón “Iniciar conexión”, situado en la parte superior izquierda:

  - Rellenaríamos nuestro usuario y contraseña y pulsaríamos acceder:

- Acceder al listado de plantillas a través de la opción plantillas del menú.



- Marcar plantilla como seleccionada., accediendo a la plantilla deseada con el botón “edit” y rellenando el campo seleccionada a 1 y pulsando guardar:

- Desmarcar plantilla como seleccionada., accediendo a la plantilla anteriormente seleccionada con el botón “edit” y rellenando el campo seleccionada a 0 y pulsando guardar.

5 SELECCIÓN DE ESTILO.

Los estilos que presenta el CMS son instalables y seleccionables, el proceso de instalación es el mismo que el de cualquier otro componente y para su selección se deben seguir los siguientes pasos:

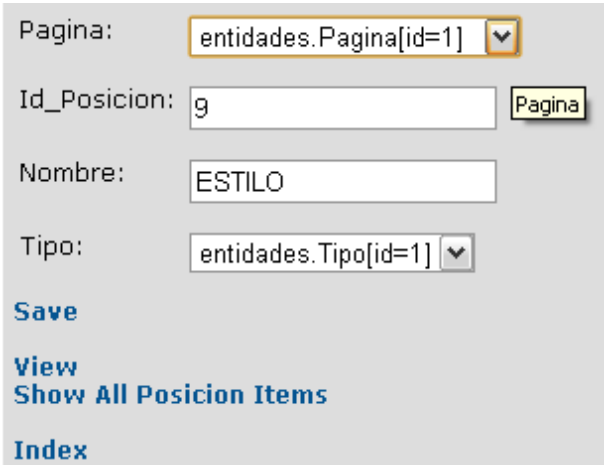
- Deberemos loguearnos en la aplicación, para ello y suponiendo que este activo el componente por defecto para la gestión de usuarios:
  - Pulsaremos sobre el botón “Iniciar conexión”, situado en la parte superior izquierda:

  - Rellenaríamos nuestro usuario y contraseña y pulsaríamos acceder:

- Acceder al listado de posiciones a través de la opción poción del menú.

- Seleccionar posición de estilo asociada a la plantilla actual.

- Editar pagina asociada a posición del listado que nos ofrece y pulsar guardar:



6 SELECCIÓN DE COMPONENTES.

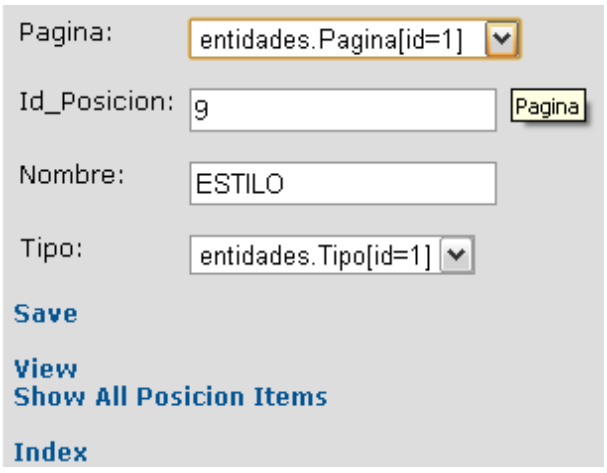
Los componentes que presenta el CMS son instalables y seleccionables, el proceso de instalación esta descrito anteriormente y para su selección se deben seguir los siguientes pasos:

- Deberemos loguearnos en la aplicación, para ello y suponiendo que este activo el componente por defecto para la gestión de usuarios:
  - Pulsaremos sobre el botón “Iniciar conexión”, situado en la parte superior izquierda:

  - Rellenaríamos nuestro usuario y contraseña y pulsaríamos acceder:

- Acceder al listado de posiciones a través de la opción poción del menú.

- Seleccionar posición a la que queremos asociar el componente.
- Editar pagina asociada a posición del listado que nos ofrece y pulsar guardar:



7 ANEXO A: INSTALACIÓN DE GLASSFISH.

7.1.1.1 Objetivo:

Realizar la instalación de glassfish v3 en Windows/Linux/Unix

7.1.1.2 Descripción:

Vamos a explicar la forma de realizar la instalación del server glassfish versión 3 en sistemas operativos Windows/Linux/Unix

7.1.1.3 Introducción.

El server glassfish v3 proporciona funcionalidad previa para la versión de Java EE 6.0 alguna de las particularides es que nos proporciona son

7.1.1.4 Pasos:

1.- El primer paso es descargar el software para el sistema operativo deseado.

http://glassfish.java.net/public/downloadsindex.html#top

2.- Ahora vamos a ejecutar la instalación del archivo que ya bajamos previamente.

Para sistema Linux/Unix tenemos que ejecutar los siguientes comandos:

```
$su – root
$chmod +x glassfish-v3-prelude-unix.sh
$./glassfish-v3-prelude-unix.sh
```

Para Windows solo es necesario hacer doble click en el archivo glassfish-v3-prelude-unix.exe

3.- Ahora solo resta aceptar la licencia y colocar algunos parámetros de configuración para proceder la instalación completa.

Puerto de administración del server: 4848

Puerto donde escucha peticiones: 8080 aunque en lo personal opino que este puerto le perteneces a tomcat por lo cual colocaremos el puerto 9090 o bien el de tu preferencia.

User: admin.

password: adminadmin

4.- Una vez que concluyo la instalación del software podrá levantar el server de una manera muy sencilla a través de una interfaz grafica haciendo click en la opción glassfish v3 prelude->Start Application–>Server. Una vez que el servidor está arriba vamos a ejecutaren un browser http://localhost:9090/

8 ANEXO B: INSTALACIÓN DE MYSQL.

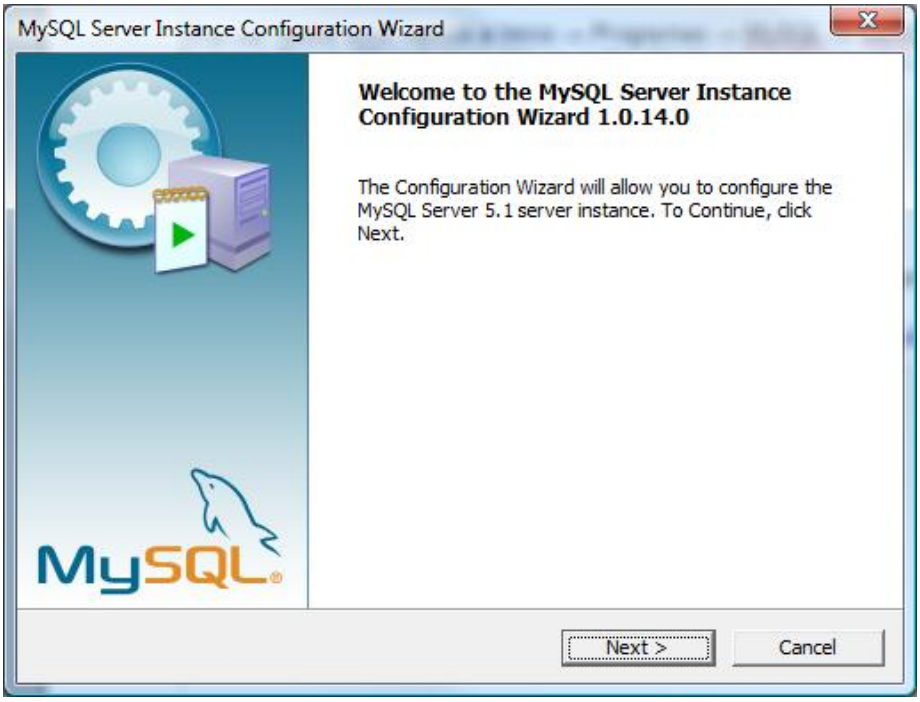
Vamos a la web oficial de MySQL y descargamos la última versión gratuita disponible llamada "MySQL Community Server".

http://dev.mysql.com/get/Downloads/MySQL-5.1/mysql-essential-5.1.31-win32.msi/from/http://mysql.easynet.be/.

Ejecutando este archivo el sistema nos muestra un asistente para la instalación del programa:

Aquí pulsamos "Next", en la siguiente pantalla seleccionamos "Typical" volviendo a pulsar "Next" y en la siguiente (si no queremos modificar la ruta donde se va a instalar) pulsamos "Install", esto provoca el comienzo de la instalación del programa, que una vez que termina muestra una pantalla donde tenemos que pulsar en "Finish".

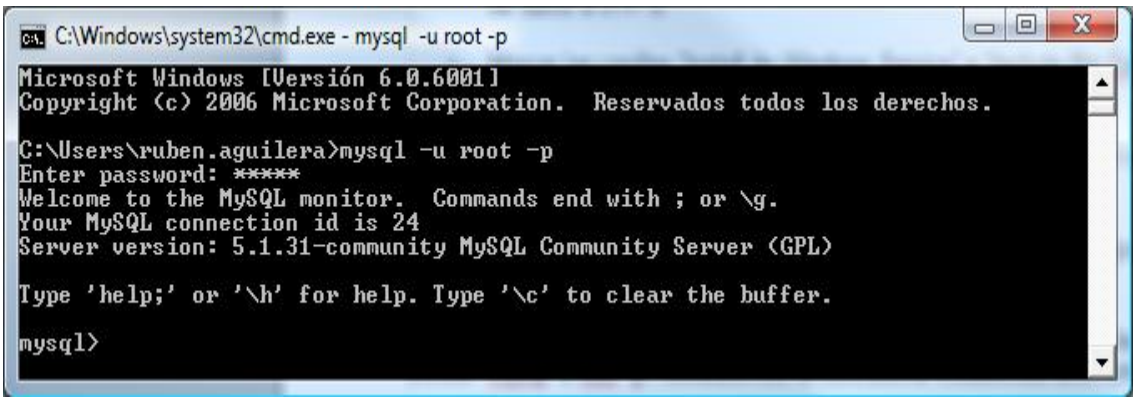
Una vez hecho esto, vamos a Inicio → Programas → MySQL → MySQL Server 5.1 → MySQL Server Instance Server Wizard (si es que no se ha abierto automáticamente).



En las sucesivas pantallas que se van mostrando al pulsar “Next” tenemos que seleccionar los siguientes datos (aunque siempre va a depender de nuestras propias necesidades):

- Detailed Configuration
- Server Machine
- Transactional Database Only
- Dejar todo por defecto
- Decision Support (DSS) OLAP
- Marcar la casilla “Enable TCP/IP Networking”, establecer “Port Number” a 3306, marcar la casilla “Add firewall exception for this port” y marcar la casilla “Enable Strict Mode”.
- Seleccionamos la opción “Best Support For Multilingualism” para establecer el encoding de la base de datos a UTF-8.
- Marcar las casillas “Install As Windows Service” e “Include Bin Directory in Windows PATH”, dejando el nombre del servicio por defecto.
- Marcar la casilla “Modify Security Settings”, estableciendo como usuario “root” y como contraseña “admin” (o la que queramos).
- Pulsamos en “Execute” para que comience el proceso de configuración y cuando finalice podemos pulsar en “Finish”.

Para comprobar que la instalación de MySQL se ha hecho correctamente podemos abrir una consola y teclear “mysql -u root -p”, introducimos la contraseña establecida anteriormente, y el sistema nos tiene que informar con una pantalla parecida a esta:



para salir de esta pantalla tecleamos “exit” y ya estaría instalado el motor de la base de datos.

ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN

Titulación:

INGENIERO EN INFORMÁTICA

Título del proyecto:

CMS-JAVA

GUIA PARA EL DESARROLLO DE COMPONENTES

Miguel Larraz Sala

Diego Pérez

Pamplona, Fecha de defensa

Control de cambios

Fecha	Autor	Descripción	Versión
22/12/2011	Miguel Larraz	Versión inicial	0.1

Revisiones

Autor	Versión	Responsabilidad	Fecha

Índice de contenidos

1 INTRODUCCIÓN.

1.1 A quien va dirigido

Este documento esta dirigido a las personas interesadas en realizar componentes para el CMS desarrollador en el proyecto, se indicaran las estructuras recomendadas para realizar estos componentes además de los recursos de los que el CMS provee.

1.2 CMS.

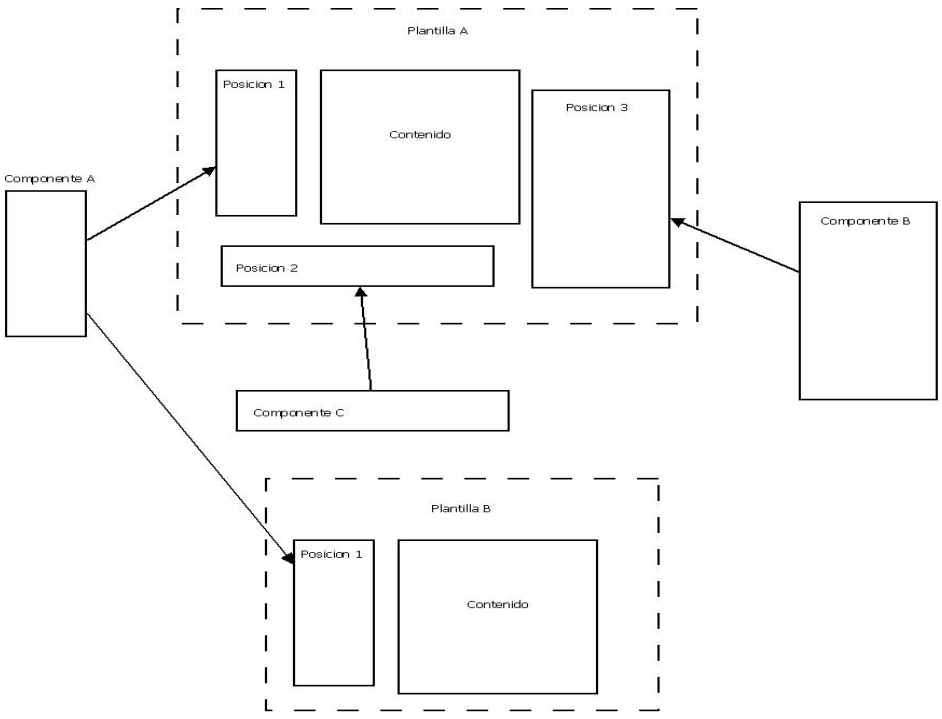
Este CMS, permite la gestión integral de los contenidos y el formato de una WEB, sin necesidad de ningún conocimiento técnico, es completamente extendible, permitiendo instalar componentes como nuevos módulos con nuevas funcionalidades, nuevas plantillas, nuevos estilos, todo esto arropado por la potencia de la plataforma JAVA.

Los componentes son una serie de paginas HTML que completan la funcionalidad del CMS, existen varios tipos de componentes unos que añaden plantillas y estilos y otros que añaden paginas y reglas de negocio, estos componentes son instalables y seleccionables, dando una completa extensibilidad al proyecto.

1.3 Estructura del CMS.

- El CMS será un contendor de plantillas, de las cuales una siempre estará activa.
- Cada plantilla tendrá además de un espacio reservado para el contenido de la pagina que se este visualizando un numero determinado de posiciones donde se podrán incrustar diferentes componentes.
- Cada componente será una pagina con sus datos y su lógica propia y completa, este componente podrá influir sobre el contenido principal de la pagina y tratar datos.
- Estos componentes se visualizaran en las posiciones que el administrador haya determinado, podrán mostrarse en varias posiciones y en diferentes plantillas.

Un pequeño esquema del CMS puede representarse de la siguiente manera.



Desde el punto de vista de la lógica de ejecución el CMS cuenta con 3 Beans que controlan la aplicación:

- **NavegacionController**: Este Bean se encarga de decidir que se va a mostrar en el navegador, su función es buscar que plantilla esta activa, que posiciones tiene esa plantilla, los componentes seleccionados para las diferentes posiciones y los destinos accesibles desde cada botón de la página.
- **SessionController**: Este Bean controla las propiedades y atributos activas en cada sesión de la aplicación, el atributo mas representativo es el usuario de la aplicación pero es accesible desde cualquier componente y puede almacenar y mostrar cualquier atributo que los diferentes componentes quieran manejar a nivel de sesión.
- **AplicationController**: Este Bean controla las propiedades y atributos activas a nivel de la aplicación, es accesible desde cualquier componente y puede almacenar y mostrar cualquier atributo que los diferentes componentes quieran manejar a nivel de sesión.

Además de estos Beans cada componente puede instalar su propio bean en el CMS que maneje la lógica del propio componente, o de varios componentes, estos bean pueden actuar a nivel de aplicación sesión o respuesta aunque es aconsejable que solo actúen a nivel de respuesta, almacenando los diferentes atributos en los beans controladores.

2 TIPOS DE COMPONENTES.

2.1 Plantillas.

Este tipo de componentes consisten en una serie de plantillas seleccionables para el proyecto CMS, consisten en una plantilla HTML, un Script SQL que registra la pagina en el listado de páginas del CMS, y si es necesario uno o varios ManagedBean, que controlen propiedades especificas de la plantilla.

2.2 Estilos.

Este tipo de componentes consisten en una serie de estilos CSS seleccionables para el proyecto CMS, consisten en uno o varios CSS, un Script SQL que registra los CSS en el listado de paginas del CMS, y si es necesario uno o varios ManagedBean, que controlen propiedades especificas del CSS.

2.3 Componentes posicionales.

Este tipo de componentes son aquellos que más tarde se pueden ubicar en alguna de las posiciones de la plantilla. Consisten en un HTML principal que será el que se posicione, un managedBean o varios que controlen las reglas de negocio, otra serie de HTML navegables a través del espacio de contenido del CMS y un Script SQL que registre el componente y si es necesario modifique adecuadamente la base de datos.

2.4 Componentes funcionales.

Este tipo de componentes son aquellos que no se posicionan en la plantilla pero que son accesibles a través del menú de componentes del CMS (menú ubicado inicialmente en el index pero que es posible invocar desde cualquier componente). Consisten en uno o varios HTML, un managedBean o varios que controlen las reglas de negocio y un Script SQL que registre el componente y si es necesario modifique adecuadamente la base de datos.

3 TECNOLOGÍAS COMPATIBLES.

3.1 Introducción.

Este CMS, tiene como particularidad, admitir la instalación de componentes que usen diferentes tecnologías, a continuación enumeramos las tecnologías que el CMS admite para el desarrollo de los componentes.

3.2 JSF 2.0.

JavaServer Faces (JSF) es una tecnología y framework para aplicaciones Java basadas en web que simplifica el desarrollo de interfaces de usuario en aplicaciones Java EE. JSF usa XHTML como la tecnología que permite hacer el despliegue de las páginas.

JSF incluye:

- Un conjunto de APIs para representar componentes de una interfaz de usuario y administrar su estado, manejar eventos, validar entrada, definir un esquema de navegación de las páginas y dar soporte para internacionalización y accesibilidad.
- Un conjunto por defecto de componentes para la interfaz de usuario.
- Bibliotecas de etiquetas personalizadas para XHTML que permiten expresar una interfaz JavaServer Faces dentro de una página XHTML.
- Un modelo de eventos en el lado del servidor.
- Administración de estados.
- Beans administrados.

3.3 HTML.

HTML, siglas de HyperText Markup Language (Lenguaje de Marcado de Hipertexto), es el lenguaje de marcadopredominante para la elaboración de páginas web. Es usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes. HTML se escribe en forma de «etiquetas», rodeadas por corchetes angulares (<,>).

3.4 JSP.

JavaServer Pages (JSP) es una tecnología Java que permite generar contenido dinámico para web, en forma de documentos HTML, XML o de otro tipo. Las JSP's permiten la utilización de código Java mediante scripts. Además, es posible utilizar algunas acciones JSP predefinidas mediante etiquetas. Estas etiquetas pueden ser enriquecidas mediante la utilización de Bibliotecas de Etiquetas (Tag Libraries) externas e incluso personalizadas.

3.5 Servlets.

Los servlets, son objetos que corren dentro del contexto de un contenedor de servlets (ej: Tomcat) y extienden su funcionalidad. La palabra servlet deriva de otra anterior, applet, que se refería a pequeños programas que se ejecutan en el contexto de un navegador web. Por contraposición, un servlet es un programa que se ejecuta en un servidor.

El uso más común de los servlets es generar páginas web de forma dinámica a partir de los parámetros de la petición que envíe el navegador web.

3.6 Otras.

En realidad se pueden desarrollar servlets también con el resto de tecnologías admitidas por el servidor de aplicaciones que estemos usando, aunque hay que tener en cuenta, que no siempre podrán acceder a los ManagedBeans que manejan el acceso a la capa de negocio propia del CMS, ni a los EJB que manejan el acceso a la capa de datos.

4 RECURSOS DISPONIBLES EN EL CMS.

4.1 EJB Datos.

Este EJB, se puede invocar desde cualquier bean o servlet de la aplicación war. Sirve para hacer consultas y actualizaciones tanto JPQL como SQL a la base de datos.

Para ello dispone de la siguiente interfaz

```
public java.lang.Object Consulta(java.lang.String consulta, utilidades.Propiedad[]
parametros);

public boolean Modificacion(java.lang.String modificacion, utilidades.Propiedad[]
parametros);

public java.lang.Object ConsultaNoJPA(java.lang.String consulta, utilidades.Propiedad[]
parametros);

public boolean ModificacionNoJPA(java.lang.String modificacion, utilidades.Propiedad[]
parametros);:
```

Hay que tener en cuenta que las consultas devuelven un objeto que contiene un ResulSet.

4.2 EJBs de entidades.

Estos EJB dan acceso a las diferentes entidades estándar del CMS, a través de estos EJB se pueden realizar consultas y actualizaciones usando la interfaz JPA.

Su interfaz es:

```
        public void create(T entity)

        public void edit(T entity)

        public void remove(T entity)

        public T find(Object id)

        public List<T> findAll()

        public List<T> findRange(int[] range)

        public int count()
```

4.3 SesionController.

Este bean esta instanciado a nivel de sesión por lo que sus atributos son accesibles desde cualquier punto y momento de la sesión de un usuario.

Dispone de la posibilidad de loguear un usuario en la sesión, dar de alta atributos a nivel de sesión, acceder a ellos y eliminarlos.

Para ello dispone de la siguiente interfaz:

```
      public ListaPropiedades getPropiedades()

      public void setPropiedad(Propiedad propiedad)

      public Propiedad getPropiedad(String nombre)

      public Usuario getUsuario()

      public void setUsuario(Usuario usuario)
```

Para cerrar sesión de un usuario bastara con hacer un setUsuario(NULL)

4.4 AplicationController.

Este bean esta instanciado a nivel de aplicación por lo que sus atributos son accesibles desde cualquier punto y momento de la aplicación.

Dispone de la posibilidad de dar de alta atributos a nivel de aplicación, acceder a ellos y eliminarlos.

```
public ListaPropiedades getPropiedades()

public void setPropiedad(Propiedad propiedad)

public Propiedad getPropiedad(String nombre)
```

4.5 NavegacionController.

Este bean esta instaciado a nivel de sesion y controla la carga dinámica de paginas plantillas estilos y destinos.

Dispone de la siguiente interfaz:

```
public String getContexto()

public void setContexto(String Contexto)

public String doDameComponente(String token)

public String getEstilo()

public void setEstilo(String estilo)

public String dameDestino(String token, String nombrePagina)

public String dotoIndex()

public String getDir_Plantilla()

public void setDir_Plantilla(String dir_Plantilla)
```

Con esta interfaz debemos montar todas las paginas y navegaciones de nuestros componentes si queremos que sean dinámicos.

5 CREAR PAGINAS Y PLANTILLAS.

5.1 Paginas.

Hay que tener en cuenta que todas las plantillas XHTML de JSF tienen que contener la siguiente estructura para que estén integradas con los estilos de JSF:

```
<ui:composition xmlns:ui="http://java.sun.com/jsf/facelets"
                template="#{navegacionController.plantillaActual.dir_Plantilla}"
                xmlns:c="http://java.sun.com/jsp/jstl/core"
                xmlns:h="http://java.sun.com/jsf/html"
                xmlns:f="http://java.sun.com/jsf/core">

                   ///CONTENIDO SEGÚN PLANTILLAS COMPATIBLES///
                                 //POR DEFECTO://
<ui:define name="content">

</ui:define>
                   ///CONTENIDO SEGÚN PLANTILLAS COMPATIBLES///
</ui:composition>
```

5.2 Plantillas.

En las plantillas que creemos debemos siempre indicar el estilo de esta manera:

```
<link href="#{navegacionController.estilo}" rel="stylesheet" type="text/css" />
```

Para crear componentes en la plantilla lo haremos de la siguiente manera:

```
            <div id="top">
                <ui:include src="#{plantillaGeneralComponent.top}"/>
            </div>
```

Y siempre deberemos incluir un apartado para el contenido, por ejemplo:

```
<div id="content" class="right_content">
                    <ui:insert name="content">Content</ui:insert>
</div>
```

6 ESTRUCTURA DEL COMPONENTE.

El componente siempre será un ZIP que contendrá lo siguiente:

- Paginas HTML, XHTML, JSP, HTML.

  Estas Páginas se colocaran en el directorio de paginas de componentes y seran accesibles para las plantillas o para el contenido.

- Servlets Clases, Beans compilados.

  Estos objetos serán insertados en la aplicación y tras el reinicio de la misma estaran disponibles para su uso e instanciacion.

- Script SQL llamado: instalacion.sql.

  Para poder instalar paginas a través de un componente hay que registrarlas en la base de datos para que los sistemas dinámicos puedan tener constancia de su existencia, esto también pasa con las plantillas, para dar soporte a esto se puede crear un instalacion.sql en el componente que registre en la base de datos estos datos durante la instalación.

  Además podemos usar este script para dar de alta campos y tablas necesarios para nuestro componente y cargarlos de datos.

- Otros.

  El resto de los objetos se quedaran en la carpeta resources/images donde podrán ser accedidos por la aplicación.