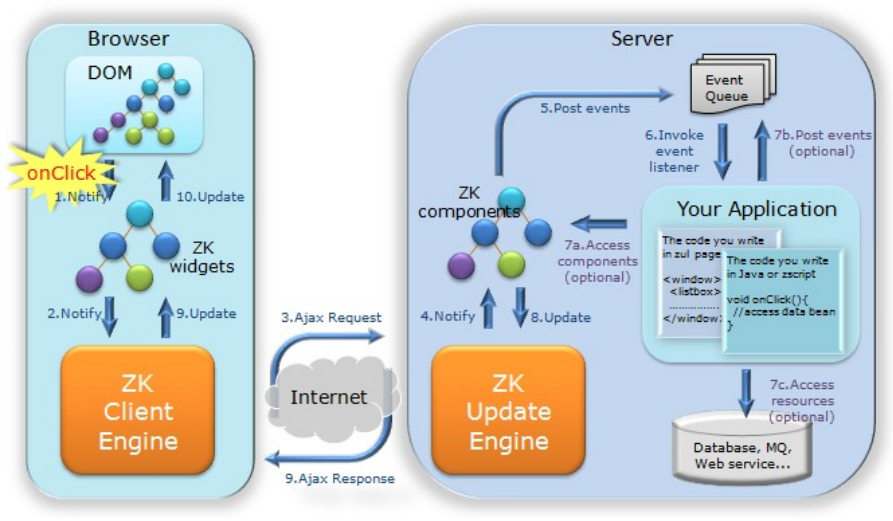
INFORMACION DE LA ARQUITECTURA DEL SISTEMA

**2. ARQUITECTURA DEL SISTEMA**

**2.1 Plataforma Java, Framework ZK**

La arquitectura planteada en el **Sistema Tracking** está basada en gran parte en la plataforma Java con framework denominado ZK que es una tecnología de muy nueva en el campo de elaboración de aplicaciones web. La cual plantea muchísimas mejoras con respecto a la plataforma Java JEE que es el modelo común de desarrollo. Esto unido a la interacción de eventos, hace que la nueva plataforma Java con un framework sea mucho más fácil de utilizar para el desarrollo de aplicaciones empresariales complejas y muy muy intuitivo para el usuario final.

Desde un punto de vista general la arquitectura sugerida se puede describir de la siguiente manera:

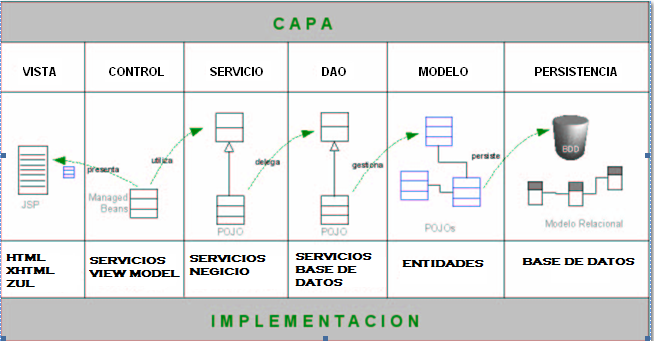


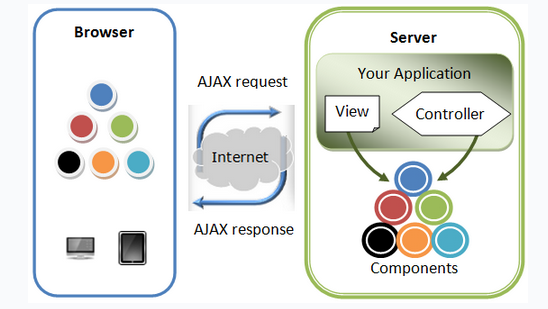
El framework ZK se ejecuta como un conjunto de servlets, dentro de un contenedor de servlets Java. El objetivo principal de la configuración del framework, es preparar el ambiente en donde las aplicaciones ZK se desarrollaran y ejecutara. Para esto se requiere la instalación de ciertos programas fundamentales para el funcionamiento del framework, esto incluye la instalación de Java Runtime Environment, la instalación de un contenedor de servlets Java y un servidor para desplegar los archivos War.

El contenedor Web es el responsable de construir la capa de presentación que será entregada a los clientes y de ejecutar las acciones de negocio utilizando los componentes existentes. Esta tecnología está basada en un modelo no muy común y se lo conoce como Modelo-Vista-Vista-Controlador. Que se diferencia del modelo tradicional EJB que es un contenedor de servicios. El modelo del framework ZK nos permite tener un control más amplio de la lógica de negocios.

**2.2 Arquitectura por capas**

La arquitectura de la aplicación se puede dividir en varias capas lógicas que tienen su correspondiente implementación utilizando la especificación JEE 5, como se muestra en la siguiente figura:





Esta arquitectura por capas lógicas se resume de la siguiente manera:

**1 Persistencia base de datos:**

La capa de persistencia describe el mecanismo para obtener e insertar información de la base de datos. La persistencia de datos es realizada mediante un mapeo Objetos-Relacional que nos permite almacenar un modelo de objetos en un modelo relacional utilizando tablas de base de datos, en forma transparente y sin tener que utilizar sentencias SQL. Dicho mapeo permite expresar las relaciones entre objetos de un modelo de objetos como relaciones entre tablas utilizando llaves foráneas o tablas para romper relaciones.

La capa de persistencia es uno de los componentes más importantes de la arquitectura, puesto que facilita el desarrollo de aplicaciones y permite ignorar los detalles específicos del mapeo de datos a una base de datos.

**2 Servicio de base de datos:**

El Modelo representa todos los conceptos del dominio de negocio que necesitemos modelar dentro de la aplicación. Por ejemplo, objetos de nombre Dispositivo, Parámetro, Estructura Organizacional etc. Representa el modelo para desarrollo con la tecnología ZK.

La capa Modelo es fundamental en toda la arquitectura, puesto que son estos objetos que representan el estado del sistema los que serán llevados de capa en capa, hasta poder presentarlos al usuario.

**3 Entidades (DAO):**

La capa DAO permite desacoplar el mecanismo de persistencia de datos de la capa de negocio (Servicio), para poder intercambiar dicho mecanismo de persistencia en el futuro.

**4 Servicio View Model:**

Esta es la capa más importante de la arquitectura pues define el comportamiento del sistema, y engloba todas las posibles funcionalidades del mismo. Es una buena práctica definir la capa de servicio como una interface, para desacoplar la estrategia de implementación de la definición de negocio. En la arquitectura sugerida las interfaces de Servicio son implementadas como Session ZK, logrando con ello tener una transaccionalidad y seguridades declarativas.

**5 Control**

La capa de control gestiona las peticiones de los clientes que ingresan al sistema. Es en esta capa donde comienza toda la interacción con el cliente. La capa de Control utiliza la capa de Servicio para solucionar los requerimientos de negocio

**6 Vista**

Esta capa es la última de la arquitectura y contiene los mecanismos de presentación de interface de usuario. En el caso de la arquitectura sugerida, esta capa puede estar formada por componentes ZK dentro de páginas JSP. Si se quisiera pasar a otro tipo de componentes, esta es una cuestión de implementación que no afecta en la organización arquitectónica de la aplicación, ya que se siguen respetando los patrones y capas definidos anteriormente.

**3.3 Tecnologías Utilizadas en Cada Capa.**

CAPA PERSISTENCIA: para el mapeo objeto relación se utilizara Eclipselink comprendida con la siguiente especificación.

* Java Persistence API 1.0
* Common Annotations for the Java Platform 1.0

CAPA DAO

* Java Message Service (JMS) 1.1
* Java Transaction API (JTA) 1.0

CAPA DE SERVICIOS

* Propias de la implementacion es muy libre en su implemantacion.

CAPA DE CONTROL

* ZK
* Servlet 2.4
* JavaServer Faces 1.2 Requirements

CAPA VISTA

* zul
* JavaScript

**3.4 Herramientas Utilizadas Para el Desarrollo e Implementación**

Las herramientas que se utilizara para el desarrollo del sistema son:

ID DE DESARROLLO

* Netbeans 7.3.1
* Plug-in ZK Para Netbeans

SERVIDOR DE APLICACIONES

* Gasfish 4.0

BASE DE DATOS

* MaySQL 5.0
* PostgresSQL

**3.5 Requerimientos de Hardware y Software.**

* Base de datos MySQL 5.0
* Servidor Glasfish 4.0
* Ireport 4.5.1
* Power Designer 16.0.
* 2 Máquina de desarrollo Hp I7 16 en RAM, 1TB de disco duro. (Proveedor)
* Navegador Google Chrome, Mozilla, etc.
* Contenedor de intercambio de información.
* Máquina virtual VMWare.