Capítulo III. Análisis y diseño.

3.1 Análisis.

El análisis es el intermediario entre los requisitos del sistema y el diseño, esta sección definiremos el análisis con una serie de modelos técnicos del sistema, utilizando formatos en texto y diagramas para representar los requisitos. El análisis se centra en lo que se debe de hacer para cumplir con lo planteado en la propuesta formal y no en el cómo, se hará uso de la ingeniería web para definir los elementos de análisis.

3.1.1 Diagrama de flujo de datos.

El diagrama de flujo de datos se divide en dos; (1) el diagrama de contexto muestra de una forma general el uso de la aplicación, es decir el proceso principal y la dirección de flujo de información. (2) El diagrama de nivel superior o diagrama UML muestra el flujo de datos de una manera más detallada, muestra las ligas a cada página de la aplicación, operaciones con java script y el direccionamiento después de un submit.

3.1.1.1 Diagrama de contexto (Nivel 0).

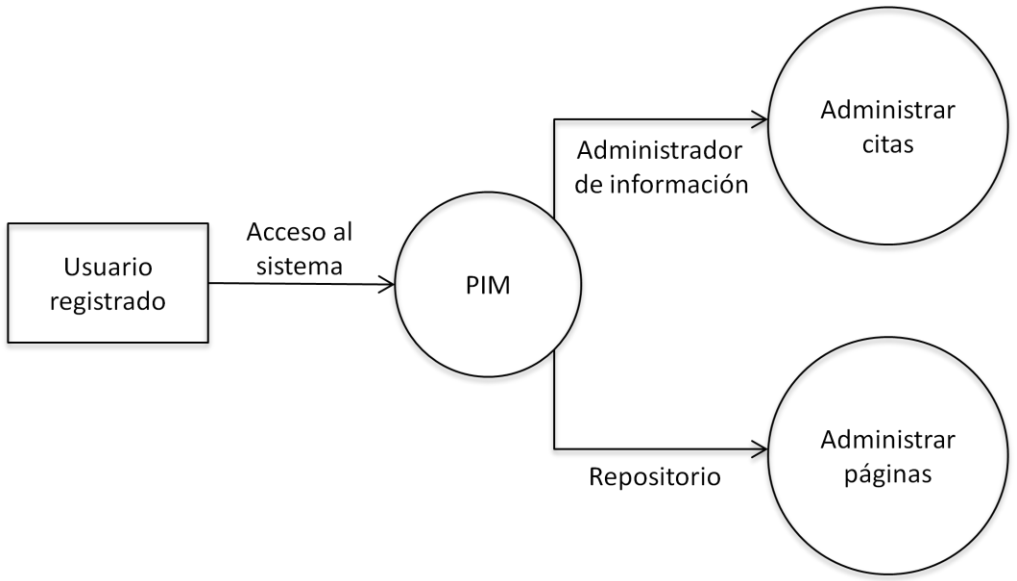

Figura 1. Diagrama de contexto.

3.1.1.2 Diagrama de nivel superior (Nivel 1).

Este diagrama fue realizado en base al diagrama UML para aplicaciones web.

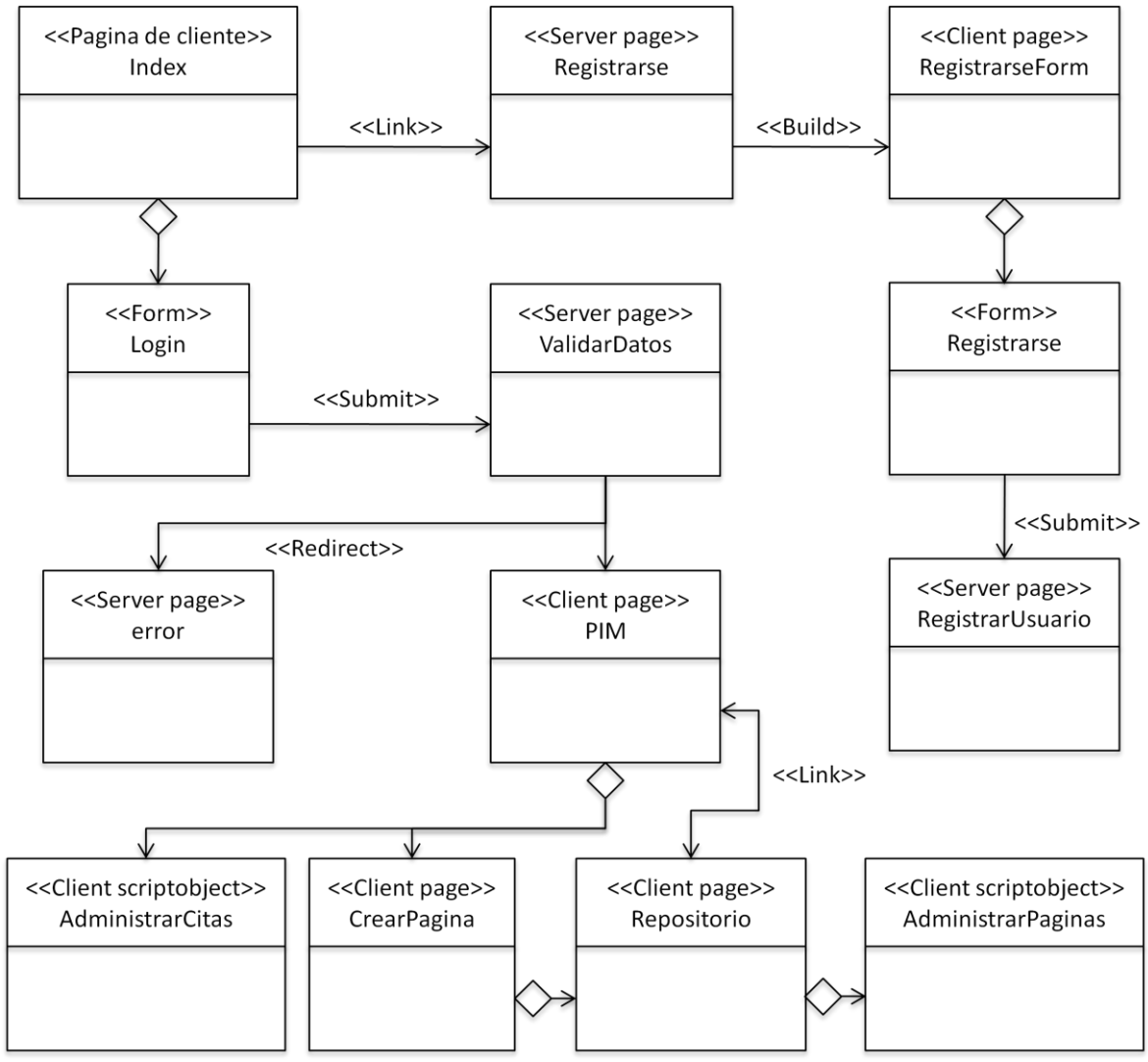

Figura 2. Diagrama de nivel superior.

3.1.2 Casos de uso.

En este apartado definiremos los casos de uso, estos representan de una forma grafica la relación del cliente con el sistema, para la aplicación solo tenemos un usuario, el que se

registra y hace uso e interactúa de una manera única con la aplicación y la información correspondiente a la sesión del usuario.

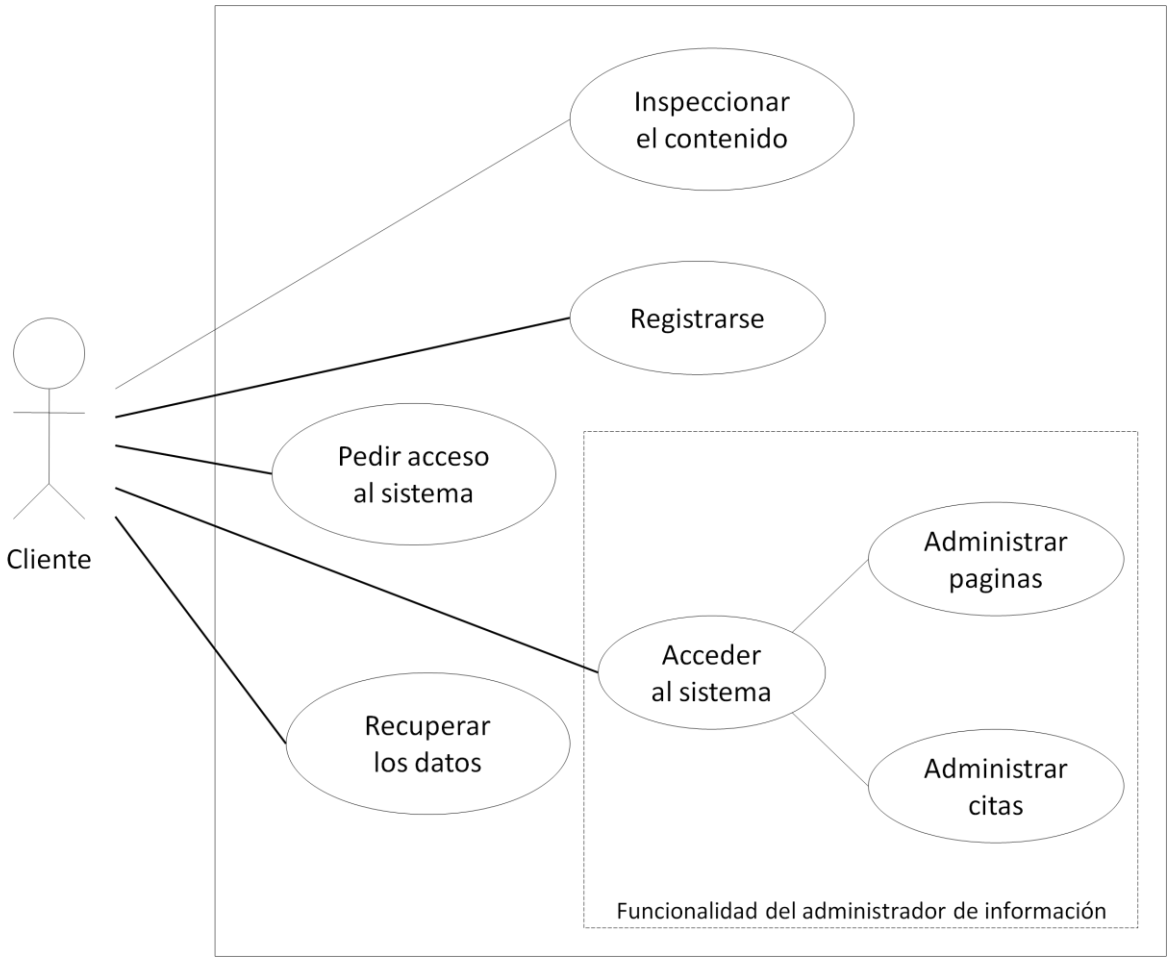

Figura 3. Diagrama de casos de uso.

3.1.3 Modelo de datos conceptual.

La siguiente figura muestra el diseño de la base de datos, con sus respectivas restricciones y operaciones de manipulación.

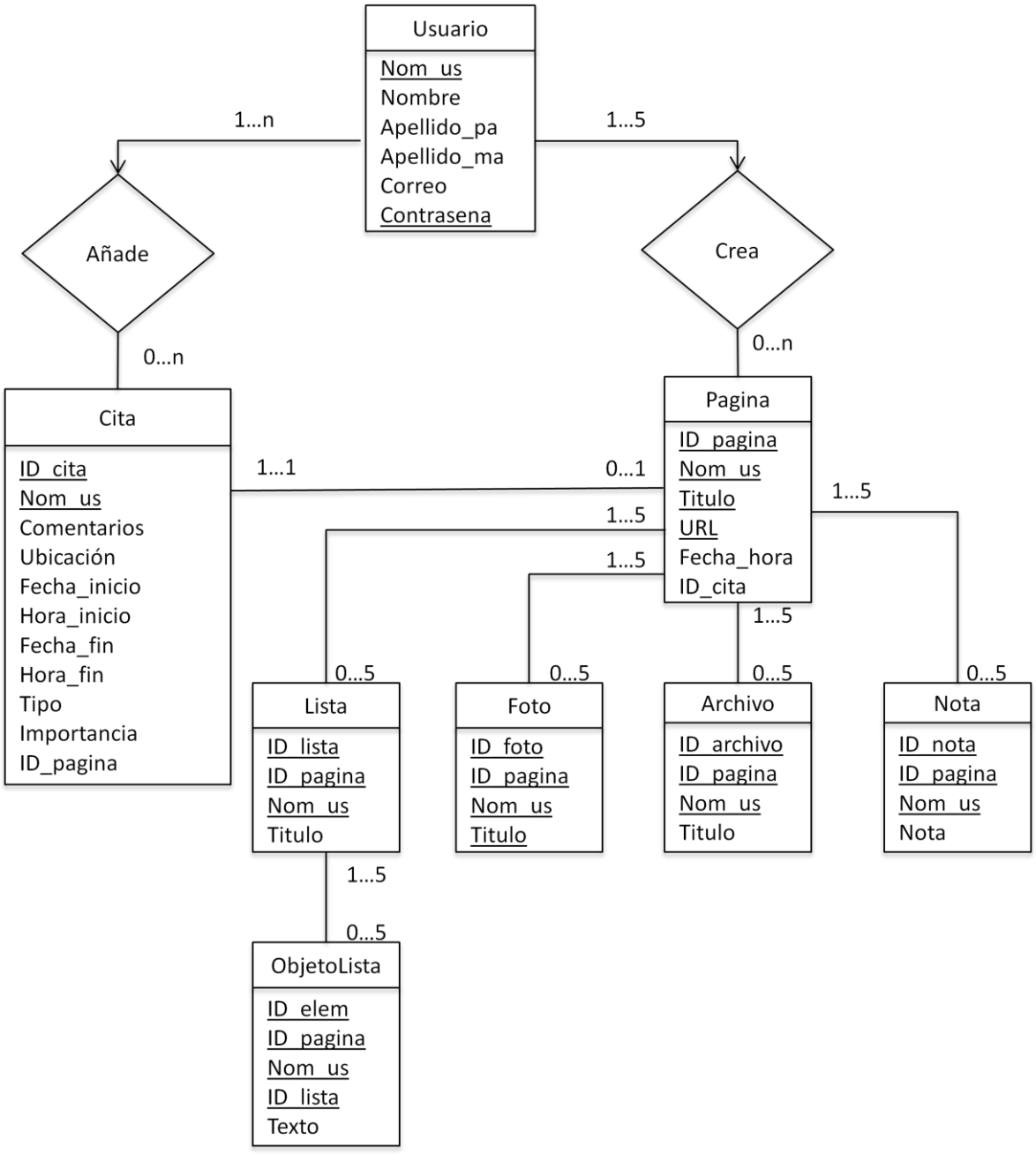

Figura 4. Modelo de datos.

3.2 Diseño

El diseño del sistema se realizo en base a la pirámide de diseño para la ingeniería Web basada en el libro de Pressman[10]. Cada elemento de la pirámide denota una actividad que

se realizo durante el diseño del administrador de información. Los siguientes puntos son cada uno de los elementos de la pirámide.

3.2.1 Interfaces

Las interfaces fueron diseñadas de la manera más sencilla, para una navegación fácil y entendimiento básico y sencillo para el usuario. A continuación se muestran las imágenes de los posibles diseños de las interfaces que compondrán al sistema.

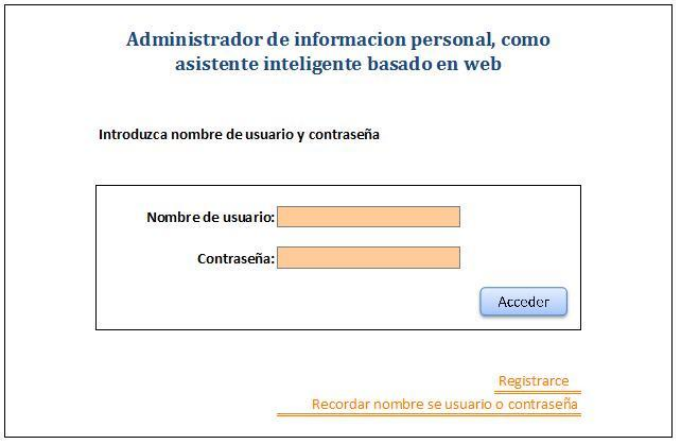

Figura 5. Acceso al sistema.

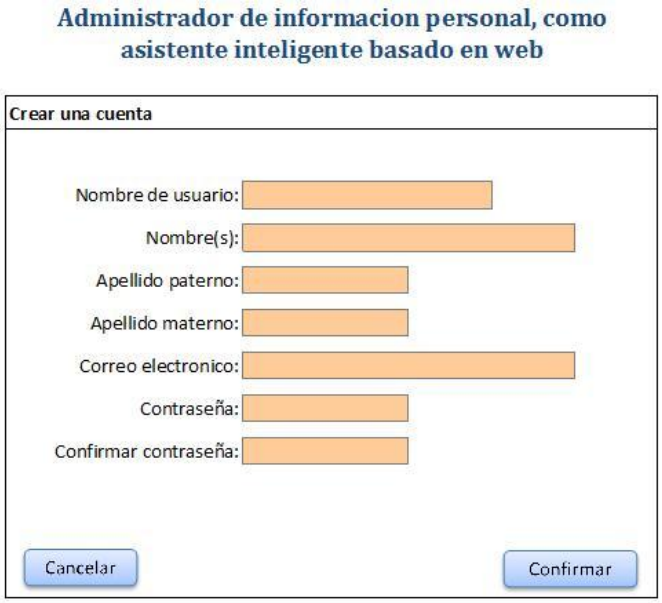

Figura 6. Registro de clientes para acceder al sistema.

Figura 7. Administrador por día.

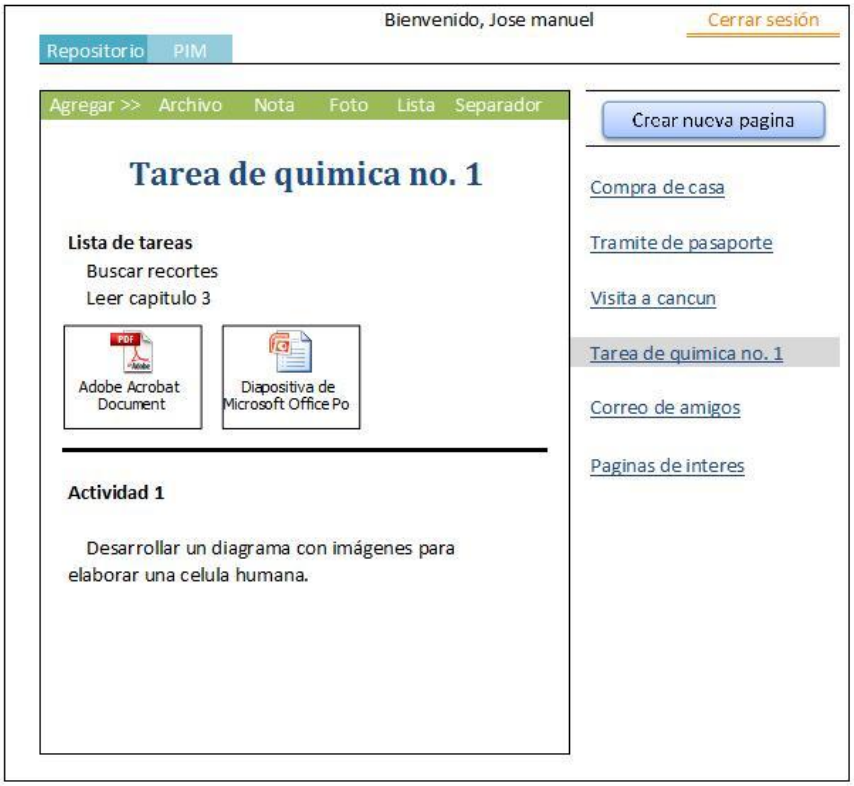

Figura 8. Repositorio.

3.2.2 Estética.

El diseño estético comprende las hojas de estilo que utilizara la aplicación, donde se definen colores, fondos, fuentes, plantillas y ubicación de los elementos de las páginas. Las páginas de la aplicación tendrán un patrón en cuanto a estilo, para facilitar el re uso de las mismas sobre toda la aplicación.

3.2.3 Navegación.

La navegación muestra la relación de los componentes o páginas que interactúan en la aplicación, indica la existencia de ligas de una página a otra, la siguiente imagen muestra de una manera grafica la relación entre las diferentes páginas.

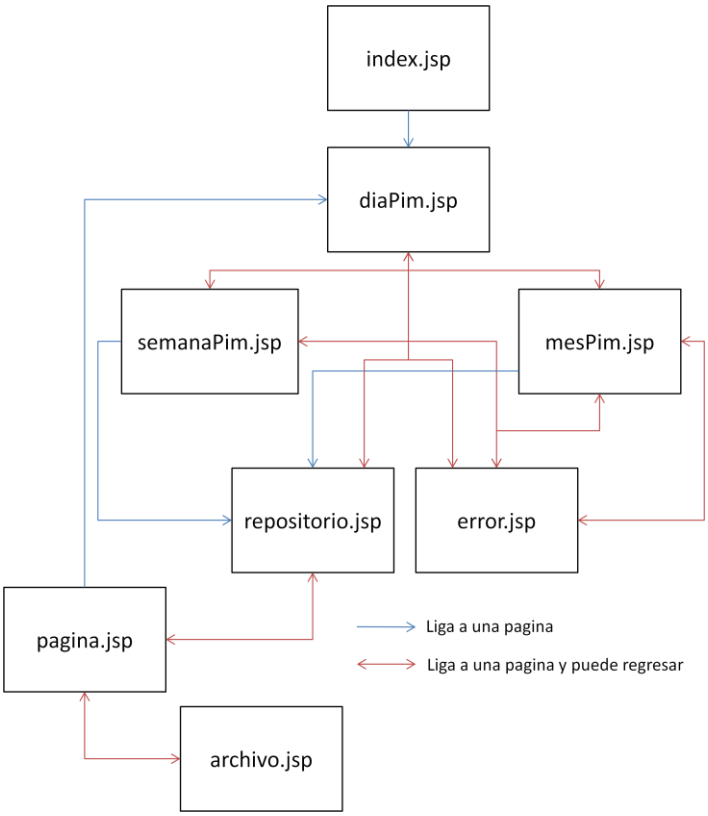

Figura 9. Diagrama de navegación.

3.2.4 Arquitectura

La arquitectura es una parte fundamental del diseño, de aquí parte como está organizada la aplicación y en donde encajan cada uno de los componentes y tecnologías a utilizar. En

esta parte especificaremos los diferentes componentes que organizaran el administrador de información con ayuda de diagramas e imágenes. Para dar un ejemplo más claro de cómo está conformada la arquitectura se pretende explicar las partes del patrón de diseño MVC (Por sus siglas en inglés Model View Controller) que fue implementado para el desarrollo de la aplicación, de tal forma que nos permita separar la vista del modelo y ser manejada por una tercera capa que es el controlador. Posteriormente se hablara de la arquitectura general del sistema con cada componente integrado.

3.2.4.1 Componentes de la arquitectura

Como parte del diseño de la vista definiremos los componentes que integran las interfaces gráficas; (a) el contenedor web es el encargado de enviar las peticiones desde la interfaz al servidor y generar las respuestas, (b) Ajax un conjunto de tecnologías que permiten la interactividad de páginas sin necesidad de recargarlas nuevamente y por ultimo (c) los JSP que son páginas dinámicas para peticiones que requiera recargar una página (Por ejemplo el acceso mediante una contraseña y nombre de usuario). Aunque Tomcat abarca parte de los tres elementos del patrón MVC, el uso será descrito desde el principio.

Apache es un servidor web robusto, para el desarrollo de esta tesis se utilizó Tomcat, un contenedor web que nos proporcionó lo necesario para el desarrollo de la aplicación. Un contenedor web es una aplicación java que controla los servlets debido a que estos no tienen un método Main como las aplicaciones comunes. El contenedor es el encargado de darle al servidor las peticiones y enviar las respuestas y llama a los métodos como doPost() y doGet() (Los encargados de hacer las peticiones al servlet).

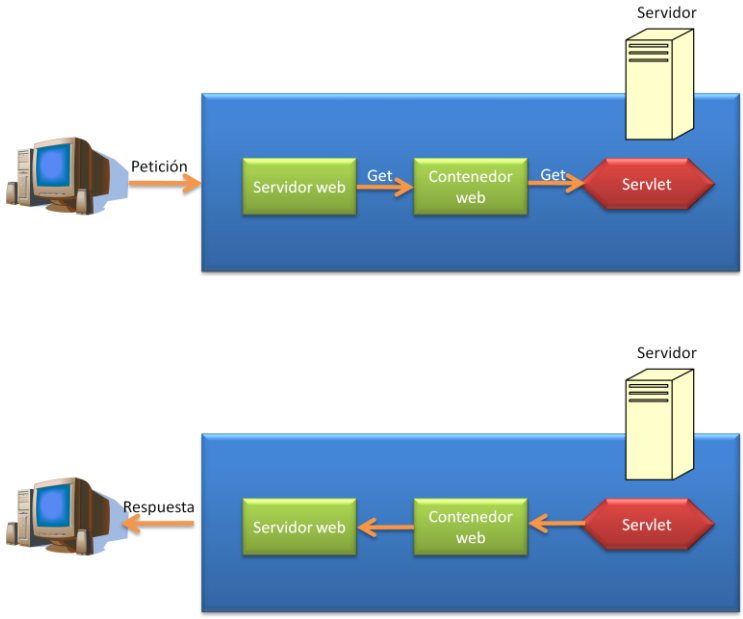

Figura 10. Petición y respuesta de un servidor.

Sin un contenedor web tendríamos que programar a bajo nivel para crear un socket de conexión con el servidor web. Entre las ventajas que ofrece usar un contendor están las de soporte de comunicación con los servidores, manejo del ciclo de vida de los servidores, soporte para múltiples peticiones que recibe un servidor (multithreading), manejo de la seguridad usando el descriptor XML y soporte para la realización de páginas dinámicas JSP o interactivas con Ajax.

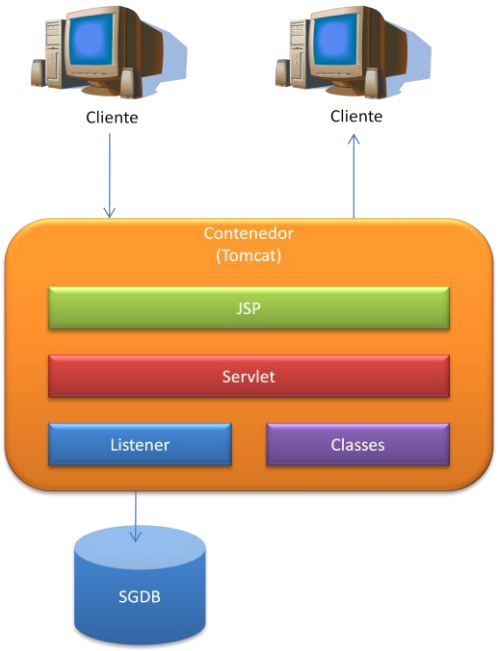

Figura 11. Funcionamiento básico de Tomcat.

Ajax es otro componente de la vista que nos permite hacer páginas interactivas sin necesidad volver a recargar la página, Ajax no es una tecnología sola, si no un conjunto de ellas como HTML, DOM (Por sus siglas en inglés *Document Object Model*), XML y XLMHttpRequest. A continuación se describen brevemente algunas de estas tecnologías.

DOM es una plataforma que proporciona un conjunto estándar de objetos a través de la cual se pueden crear documentos HTML y XML, navegar por su estructura y, modificar, añadir y borrar tanto elementos como contenidos [11]. Con esto podemos interactuar dinámicamente con las páginas web accediendo con un leguaje de scripting al servidor y recibir una respuesta en forma de texto plano o XML, para generar respuestas a las peticiones.

XML no es más que una manera de definir un lenguaje y su gramática para alguna necesidad en especifico, en Ajax XML es usado para la transferencia de vuelta al servidor. XMLHttpRequest es una interface para la comunicación asíncrona con el servidor y realizar

peticiones http en segundo plano. En la siguiente imagen se muestran la diferencia entre la arquitectura de una página web clásica y otra que implementa Ajax.

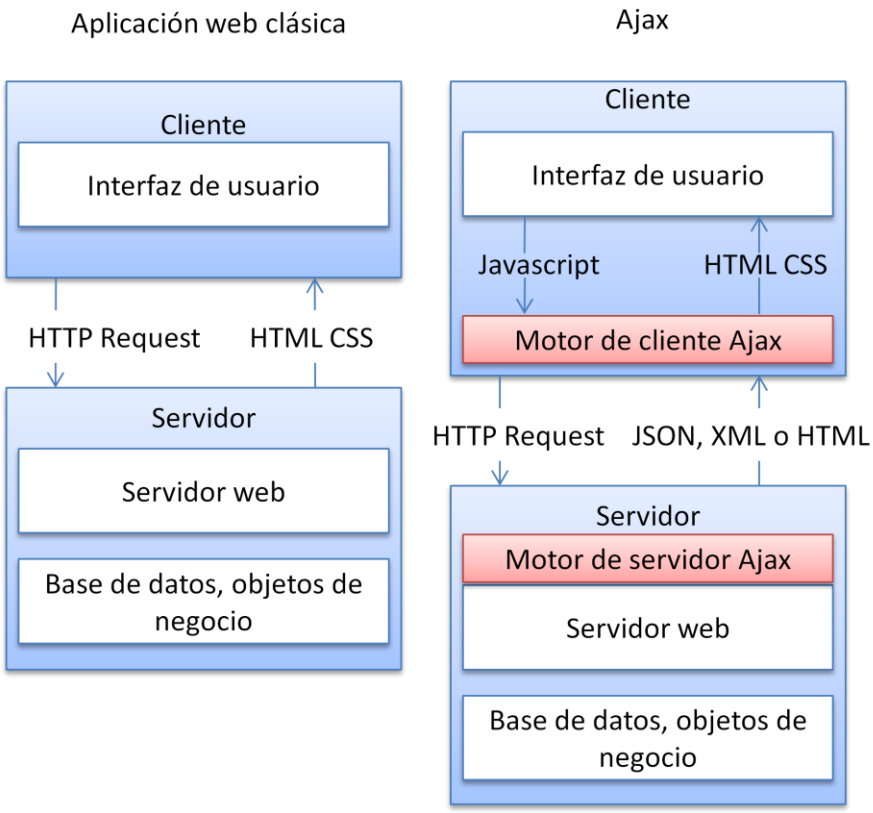

Figura 12. Arquitectura AJAX.

JSP (Por sus siglas en ingles *Java Server Pages*) es una tecnología que nos permite crear páginas dinámicas en forma HTML, es una página web con etiquetas HTML y extensión .jsp que permite incrustar código java haciendo uso de scriptlet <%...%> en donde el código java se escribe adentro de esta etiqueta.

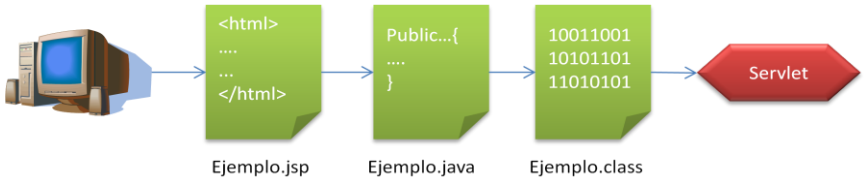

Figura 13. Funcionamiento básico de un JSP.

El cliente hace una petición sobre el JSP, el contenedor se encarga de compilar y correr las clases que se utilizan dentro del JSP y envía la petición al servlet para generar una respuesta. Los elementos que conforman el controlador y modelo de la aplicación serán explicados a lo largo de esta sección (Arquitectura).

Spring provee una manera fácil de manejar los objetos de negocio. Las capas de la arquitectura de Spring pueden usarse por separado aislándolas una de otra, por ejemplo Spring puede usarse para simplificar el uso del JDBC (Pos sus siglas en ingles *Java Database Connectivity*) o bien para administrar todos los objetos de negocio de la aplicación. En este caso será utilizada la segunda alternativa.

Los servlets son objetos que corren dentro del contexto de un contenedor de servlets (ej: Tomcat) y extienden su funcionalidad [12]. Los servlets son programas escritos en Java que se ejecutan de lado del servidor en una aplicación web. Los servlets son clases que implementan la interfaz javax.servlet.Servlet.

Como se menciono anteriormente los JSP (Un caso especial de servlet) permiten generar contenido dinámico. Con ayuda de un servlet, las peticiones que se envían desde un JSP podrán ser recibidas por los servlets y procesarlas para generar una respuesta. Los objetos HttpServletRequest y HttpServletResponse representan las peticiones y las respuestas respectivamente de la página que invoco al servlet.

Hibernate es una herramienta de Mapeo objeto-relacional para la plataforma Java que facilita el mapeo de atributos entre una base de datos relacional tradicional y el modelo de objetos de una aplicación [13]. Hibernate elimina el tedioso y repetitivo trabajo de codificar y permite a los desarrolladores enfocarse en los problemas de negocios [14].

Hibernate ayudara a la aplicación a crear una base datos con los objetos ya existentes de nuestra aplicación, de esta manera se pueden guardar objetos con solo usar

session.save(miObjeto) para el almacenamiento de citas, eventos y los elementos almacenados en el repositorio. En la parte del modelo se encuentran las clases que se conectan a la base de datos, este parte se puede apreciar mejor en el diagrama de la arquitectura tres capas.

3.2.4.2 Arquitectura de tres capas

La arquitectura tres capas como su nombre lo indica se divide en tres partes, la capa de presentación o interfaz de usuario, la capa de negocio y la capa de acceso a datos. Cada capa cuenta con varios componentes esenciales para el buen funcionamiento, entendiéndose por componente a: Una de las partes de la solución total como los componentes de software compilado y otros elementos de software, como páginas web [15]. Estas capas cumplen con una función en específico que serán explicadas a continuación.

Figura 14. Arquitectura general

La capa de presentación es una forma de interactuar con el usuario, en este caso está formada por las páginas web y JSP. Es la única parte que puede ser visualizada por los usuarios. Permite procesar y dar formato a los datos de los usuarios, así como adquirir y validar los datos entrantes procedentes de éstos [15].

La capa de negocio está compuesta por los objetos que son administrados por la aplicación, como las citas, usuarios y páginas para el repositorio. Son las clases simples que serán almacenadas por la base de datos, además de componer la parte lógica de la aplicación encargada de hacer las operaciones o transacciones y generar un resultado. Esta capa representa el controlador y modelo del patrón de diseño MVC, donde los servlets son los encargados de recibir la información, procesarla y enviar la respuesta a la vista.

"La mayoría de las aplicaciones y servicios necesitan obtener acceso a un almacén de datos en un momento determinado del proceso empresarial" [15]. La capa de acceso a datos permite lograr esta función, en esta parte se encuentran las clases que crean la conexión, transacciones, peticiones y registros con la base de datos de una forma transparente para la capa de negocio.

Oracle será usado como gestor de base de datos en la versión 10g, permitirá la persistencia de los datos. La interface ServletContextListener del api de Java permite crear una conexión con la base de datos de Oracle, y crear código más limpio, la clase que implementa esta interface se da de alta en el descriptor XML con la etiqueta <listener>. Sin embargo para esta tesis se utilizó la clase Conexión creada en proyectos anteriores y con la que se tiene una familiarización.

3.2.5 Diseño de componentes

Los componentes del sistema son el modelo de datos lógico, las tablas que comprenderán la base de datos y el modelo lógico que se encarga de parte lógica del

sistemas como el almacenamiento, modificación, obtención y eliminación de los diferentes objetos de negocio usados por la aplicación.

3.2.5.1 Modelo de datos lógico.

A continuación de describen cada una de las tablas utilizadas para el desarrolla de la aplicación con sus respectivos campos y tipos.

Campo	**Tipo**
Id	Numero de dos dígitos
nom_us	Cadena de 20 caracteres
Nombre	Cadena de 20 caracteres
apellido_ma	Cadena de 10 caracteres
apellido_pa	Cadena de 10 caracteres
Correo	Cadena de 50 caracteres
Contraseña	Cadena de 12 caracteres

Tabla 1. Usuario.

Campo	**Tipo**
nom_us	Cadena de 20 caracteres
Titulo	Cadena de 30 caracteres
Comentario	Cadena de 50 caracteres
Ubicación	Cadena de 20 caracteres
fecha_inicio	Cadena de 10 caracteres
hora_inicio	Cadena de 4 caracteres
fecha_fin	Cadena de 10 caracteres
hora_fin	Cadena de 4 caracteres

Tipo	Cadena de 15 caracteres
Importancia	Cadena de 15 caracteres
Pagina	Cadena de 30 caracteres

Tabla 2. Cita.

Campo	**Tipo**
Id	Cadena de 30 caracteres
nom_us	Cadena de 20 caracteres
fecha_inicio	Cadena de 10 caracteres
hora_inicio	Cadena de 4 caracteres

Tabla 3. Página.

Campo	**Tipo**
Id	Cadena de 30 caracteres
id_pagina	Cadena de 30 caracteres
nom_us	Cadena de 20 caracteres
url_archivo	Cadena de 100 caracteres
Extensión	Cadena de 4 caracteres

Tabla 4. Archivo.

Campo	**Tipo**
Id	Cadena de 30 caracteres
id_pagina	Cadena de 30 caracteres
nom_us	Cadena de 20 caracteres
Nota	Cadena de 100 caracteres

Tabla 5. Nota.

Campo	Tipo
Id	Cadena de 30 caracteres
id_pagina	Cadena de 30 caracteres
nom_us	Cadena de 20 caracteres

Tabla 6. Lista.

Campo	Tipo
Id	Cadena de 30 caracteres
id_lista	Cadena de 30 caracteres
id_pagina	Cadena de 30 caracteres
nom_us	Cadena de 20 caracteres
valor	Cadena de 5 caracteres

Tabla 7. Objeto lista.

Campo	Tipo
Id	Cadena de 30 caracteres
id_pagina	Cadena de 30 caracteres
nom_us	Cadena de 20 caracteres

Tabla 8. Foto.

Campo	Tipo
Id	Cadena de 15 caracteres
Valor	Número de 2 decimales

Tabla 9. Tipo.

Campo	Tipo
Id	Cadena de 15 caracteres
Valor	Número de 2 decimales

Tabla 10. Importancia.

3.2.5.2 Modelo lógico.

El modelo lógico son las clases utilizadas para generar objetos; la clase cita, usuarios, páginas, archivos, notas, listas, objetos de la lista y finalmente fotos. Son los beans u objetos de negocio que serán manejados por el sistema, cada bean se relaciona con su respectiva tabla descrita en el apartado anterior. Todos lo objetos de negocio serán JavaBeans con sus respectivos métodos geters y seters, o accesores y modificadores.

Cada clase se reparte en paquetes, para el controlador tenemos los paquetes, almacenar, borrar, modificar y obtener, y las acciones para ligas repositorio, página, pim y registro. Para el modelo tenemos los beans, modelo de página, modelo de citas y modelo de usuarios, así como la conexión a la base de datos y para la vista tenemos los archivos jsp que son día, semana, mes, repositorio, página, archivo y foto, al igual que el acceso y registro.

Resumen.

El análisis y diseño nos facilitaron la implementación del sistema, nos dejó una idea más clara de la relación entre los componentes de la aplicación y la base de datos. El diagrama UML o diagrama de clases permiten tener una abstracción más detallada de las clases que serán utilizadas y desarrolladas. Nos ayudara crear una aplicación flexible y modular para el re uso en futuras aplicaciones o una posible extensión para dispositivos móviles.