Universidad de las Ciencias Informáticas
Dirección de Informatización

Facultad 10

Título: “Diseño y montaje de un cliente para la administración de un registro UDDI”

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas.

Autores: Darien García Tejo.

Rolaidis Oliver Martell.

Yubisleidy Romero Romaguera.

Tutores: Ing. Aleida Eva Sáez Aldana.

Ing. Manuel Alejandro Gil Martín.

Ciudad de la Habana
Junio de 2008

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma con carácter exclusivo.

Para que así conste firmo la presente a los__ días del mes de__ del 2008.

Darien García Tejo.

Firma del Autor

Yubisleidy Romero Romaguera.

Firma del Autor

Rolaidis Oliver Martell.

Firma del Autor

Aleida Eva Sáez Aldana.

Firma del Tutor

Manuel Alejandro Gil.

Firma del Tutor

AGRADECIMIENTOS

Agradezco de todo corazón:

A mami y a papi, por brindarme todo su apoyo, confianza, educación y amor, por creer siempre en mí, por guiarme por el mejor camino, por ser mi ejemplo y mi guía a lo largo de todo este trayecto, ustedes se merecen mucho más que esto.

A mis segundos padres Idalmis y Valerio, a quienes respeto y quiero mucho, por siempre apoyarme y principalmente en estos 5 años, siempre preocupados, atentos, por brindarme su ayuda incondicional, por quererme y hacerme parte de su familia.

A mi hermanita, a pesar de ser menor que yo, siempre ha estado ahí para lo que me ha hecho falta, por quererme, por apoyarme, y se que seguirás mis pasos.

A mi novio Dashiell, por apoyarme, por brindarme su ayuda, por sus preocupaciones, por desvelarse junto a mí día tras día, por aportar ideas para el desarrollo de este trabajo, por quererme, por soportarme y tener tanta paciencia conmigo.

A mis hermanos, por quererme, preocuparse y apoyarme tanto.

A mi tía Ana, por ayudarme, preocuparse y velar por mí en estos 5 años de la carrera.

A mi tía Martha, que en muchas ocasiones ha hecho el papel de mi mamá, por sus preocupaciones, su apoyo, sus desvelos, sus caminatas hasta mi casa para aviarme del día, hora y lugar en que sale la guagua cuando estoy de pase aunque ya yo lo sepa, por siempre estar atenta a todo lo que me concierne.

A mis abuelas, mis tíos, mis tías, mis primos y primas, mí cuñada Yani, a toda mi familia en general por ser tan unida y maravillosa.

A mis compañeros de Tesis, por ser responsables y eficientes.

A mis compañeros de aula, a mis compañeras de cuarto, por haberme soportado en estos 5 años.

A mis amigas (Daymara, Yanet, Lisandra, Raycel, Dayana, Yurima, Aliuska), que hemos pasado por mucho en estos 5 años, y aunque no nos volvamos a ver por un buen tiempo, aunque cada cual se vaya para su provincia, que sepan que siempre las tendré presente, por todo lo que hemos compartido en esta universidad, por los buenos y malos momentos, por brindarme su apoyo y su preocupación cuando existe algún problema. A Chony, Saborit y Aleida, por sus ideas, ayuda, orientaciones y consejos.

A la Revolución Cubana y a nuestro Comandante Fidel Castro, por crear esta universidad, darnos la oportunidad de estudiar en ella y hacer realidad el sueño de graduarnos, siempre seremos Tu Tropa del Futuro.

AGRADECIMIENTOS

A la UCI, por ser nuestra casa en estos 5 años, por forjarnos como profesionales, y por enseñarnos a estar cada día más conectados al Futuro y conectados a la Revolución.

A todos aquellos que se acercaron a preguntar, a indagar por el estado de la tesis.

En fin a todos los que por razones de espacio no puedo mencionar aquí pero que de una forma u otra contribuyeron tanto a mi formación profesional como al desarrollo de este trabajo. Gracias a todos.

YUBI

Agradezco de todo corazón:

En primer lugar a mi mamá y a Angelito por tener que aguantarme tanto mal humor durante estos 5 años y no dejarme solo ni un solo segundo. A mi abuela que con tanto cariño y admiración siempre ha confiado en mí, a mi tía Mercedes y a mi tío Lázaro Tejo que de cierta forma ha sido mi inspiración y guía, la razón por la cual llegue tan lejos. A mi hermanito Duniel que es insoportable pero que lo quiero mucho.

A mi tía Migdalia y mi prima Yanelis, a mi segunda madre Mayelin y su esposo Ironel por apoyarme e inspirarme, a mis amigos inseparables Julio, Victor y Osniel que tanto han lidiado conmigo en estos 5 años de la carrera.

A mis demás compañeros de aula y amigos Lemay, Lester, Serguey, Ledian, Luis M Rodriguez, Luis M Teijon, Leslier, Leonardo y muy en especial a Isobert que siempre me apoyaron y ayudaron en los peores momentos. Y a todos los que de una forma u otra contribuyeron para que hoy llegase a ser lo que soy. Gracias.

Darien.

Le agradezco a todos los que de una forma u otra contribuyeron a la realización de este trabajo, a mis compañeros de estudios, a mis profesores y tutores, en especial a mi familia.

De corazón gracias.

Rolaidis.

DEDICATORIA

De YUBI:

A Irene mi mamá ya la memoria de Francisco mi amado papá *por ser mi vida y mi inspiración.*

A Lisania *mi hermana que empieza la universidad y espero le de este mismo regalo a nuestros padres.*

A Rosi *mi sobrinita de 3 añitos, que la amo como si fuera mi hija.*

De Darien:

A mi abuelo Miguel Angel Tejo *que en paz descansa.*

A Amalia *mi abuela por todo el cariño, el apoyo y la confianza que me dio.*

A mi mama Minerva y a Angelito *por ser mi guía y mi luz.*

A mi hermano Duniel*, para que le sirva de reto y llegue a ser mejor que yo.*

De Rolaidis:

A Caridad *mi mama por el inmenso amor dado, por estar siempre presentes.*

A mis tíos y tías *por formar parte de lo que soy, ser mis otros padres.*

A la memoria de Colito *mi abuela que dios lo tenga en la gloria.*

RESUMEN

El presente trabajo se desarrolló en la Universidad de las Ciencias Informáticas, centro en el cual se desarrollan una gran cantidad de aplicaciones web, tanto para el desarrollo de la universidad como para la exportación.

Este trabajo se realiza con el objetivo de lograr un software que brinde interoperabilidad, organización y publicación de los servicios web que se implementan en las diferentes aplicaciones desarrolladas en el centro. Específicamente se trata de un cliente para cualquier registro UDDI, de forma tal que brinde a los usuarios la posibilidad de hacer búsquedas sobre los servicios que se desarrollan en la institución. Para ello se estructura en 4 capítulos donde se recoge la fundamentación teórica del trabajo, el modelo del sistema, el análisis y diseño del mismo y por último su implementación y prueba.

Se propone como solución la implementación de un sistema capaz de almacenar la información referente a los servicios web así como darlos a conocer; siguiendo los estándares internacionales del catálogo de servicios web denominado UDDI (Universal Description, Discovery & Integration, en español Descripción Universal, Descubrimiento e Integración).

ÍNDICE

INTRODUCCION

Con el desarrollo de arquitecturas como SOA (Service Oriented Architecture, en español: Arquitectura Orientada a Servicios) que ha conducido a una nueva y progresiva era de la información en el mundo actual, se ha introducido una problemática con las aplicaciones web, debido al surgimiento de problemas, entre los que se perfilan:

- Desconocimiento de la existencia de los servicios web, lo que provoca la re implementación innecesaria de estos.
- Pérdida de servicios web.
- Inexistencia de un sitio en el que estén ubicados y ordenados los servicios para su adecuada publicación y búsqueda.
- Mala categorización de los mismos.

Una solución práctica a los problemas que anteriormente se perfilan sería UDDI pues ella representa un papel importante dentro de la arquitectura orientada a servicios, y su implementación a través de servicios web. En la actualidad cada organización brinda un conjunto de servicios y es, precisamente UDDI, la forma más utilizada de ofrecerlos pues constituye el papel de centro de información sobre los mismos, permitiendo encontrarlos, conocer su interfaz y accederlos de forma teóricamente automática.
UDDI no es más que un estándar para registrar y almacenar de forma estructurada dichos servicios. A través de ella, se puede publicar y descubrir información de una empresa y de los servicios que brinda, utilizando sistemas taxonómicos estándares para clasificar estos datos y poder encontrarlos posteriormente en función de la categorización. Lo más importante de UDDI es que contiene información sobre las interfaces técnicas de los servicios de una empresa. A través de un conjunto de llamadas a API XML (eXtensible Markup Language, en español: Lenguaje de Marcado Ampliable o Extensible) basadas en SOAP (Simple Object Access Protocol), se puede interactuar con la misma tanto en tiempo de diseño como de ejecución, facilitando el descubrimiento de datos técnicos de los servicios que permitan invocarlos y utilizarlos. De este modo, sirve como infraestructura para una colección de software basado en servicios web.
Su propósito general es facilitar el descubrimiento de los servicios web tanto dinámicamente como en tiempo de diseño. Para ello provee un mecanismo que le permite navegar, descubrir servicios e interactuar con ellos de manera interactiva.
En la Universidad de Ciencias Informáticas (UCI) el considerable aumento de los servicios web disponibles para el desarrollo de aplicaciones web y la necesidad de interacción entre ellas para lograr

un mayor desarrollo y rápido intercambio, demanda un software que permita lograr la interoperabilidad, organización y publicación de los servicios web implementados en las diferentes aplicaciones de manera confiable y segura, o sea, con la demanda de servicios web que existe en la UCI, se hace necesario buscar una forma de organización y estructuración de esta nueva tecnología para un mejor rendimiento y eficiencia en sus negocios.

Los autores de la presente, forman parte de un grupo de trabajo que se ha propuesto la concepción de un catálogo de servicios web, que cumpla con las especificaciones y normas de una UDDI, adaptado a las necesidades y requerimientos de la UCI y a su vez reutilizable en otros entornos.

Dado que los servicios web serán una potente herramienta para la confección de software en la UCI, no es difícil avizorar un considerable aumento de estos. Teniendo en cuenta esta predicción, se hace necesario contar con algún tipo de herramienta que posibilite una eficiente manipulación y manejo de estos servicios. Desafortunadamente la UCI no cuenta con esta herramienta ni con la experiencia necesaria para esto.

Expuestas las razones principales que hacen imprescindible la búsqueda de una tecnología capaz de solucionar este problema, queda planteado y propuesto el presente proyecto, el cual se centra en que no existe en la UCI una interfaz de usuario para un registro UDDI que sea confiable, segura y capaz de resolver los requerimientos de una SOA.

Por lo tanto, la implementación de una UDDI, es una solución práctica a dichos problemas, pues constituye un estándar para registrar y almacenar de forma estructurada los servicios web. Entre otras cosas, UDDI debe ser capaz de controlar toda la información existente y las funcionalidades del sistema, razón por la cual es necesario realizar la implementación de una interfaz de usuario utilizando una arquitectura MVC (Model View Controller, en español: Modelo Vista Controlador).

Dada la situación anterior nos planteamos el siguiente **problema científico:** ¿Cómo lograr una eficiente interfaz de usuario para UDDI utilizando el MVC sobre una arquitectura orientada a servicios y basados en software libre para la informatización de la Universidad de las Ciencias Informáticas?

El **objeto de estudio** de este trabajo se centra en las aplicaciones UDDI como sistema para la gestión de servicios web. El **campo de acción** abarca la interfaz de usuario de las aplicaciones UDDI como sistema para la gestión de servicios web en la UCI.

Para dar solución al problema antes expuesto durante todo el desarrollo del trabajo se **defiende la siguiente idea:**

Con la implementación de la interfaz de usuario para UDDI, teniendo en cuenta los estándares internacionales y seleccionando las herramientas adecuadas, se puede controlar toda la información existente, de manera tal que se elimine la pérdida y la re-implementación de servicios, lo que posibilita

contar con un seguro y confiable catálogo de servicios web basados en SOA y en software libre y consultable además para los desarrolladores y sistemas de la UCI.

Este trabajo tiene como **objetivo general** desarrollar la interfaz de usuario de una UDDI desarrollado en software libre y de acuerdo a los estándares y especificaciones internacionales.

Entre los **objetivos específicos** se plantean los siguientes:

- Analizar el estado del arte sobre las UDDIs, las tendencias actuales, así como las herramientas más recomendables para su desarrollo.
- Establecer definiciones que cumplan con los estándares internacionales.
- Realizar un estudio sobre los Sistemas Manipuladores de Contenidos (CMS) para la realización de la interfaz de usuario para UDDI.
- Realizar un estudio completo y profundo de la interfaz de usuario para UDDI.
- Diseñar la interfaz de usuario.
- Implementar un prototipo de interfaz de usuario utilizando el CMS que más se adapte al sistema que se desea obtener.

Con el propósito de guiar, controlar, evaluar y perfilar el trabajo hacia el alcance de los objetivos trazados, se definieron las siguientes **tareas**:

- Estudio y resumen de las especificaciones de una UDDI establecidas por la OASIS (Organization for the Advancement of Structured Information Standards).
- Estudio del análisis y diseño previamente elaborado del sistema.
- Declarar los requisitos que debe cumplir el sistema.
- Análisis de cómo se encuentran en el área internacional las tecnologías que se utilizan para llevar a cabo sistemas como el que se pretende implementar.
- Selección de la metodología a utilizar para el análisis y diseño de sistemas informáticos, que faciliten la creación y garanticen la calidad del sistema que se desea implementar.
- Análisis y selección de las herramientas para llevar a cabo el proyecto y la elección de la plataforma en la que se desarrollará la aplicación.
- Documentar la información referente al análisis y diseño del sistema, así como todo lo referente al flujo de trabajo.
- Estudio de los CMS.
- Diseñar una interfaz de usuario seleccionando el CMS adecuado.

- Elección del lenguaje y tecnologías a utilizar para el diseño e implementación de la interfaz de usuario para UDDI.
- Agrupar los estándares de seguridad establecidos para los servicios web.
- Definir los protocolos que se utilizan para transportar la información.
- Obtención del prototipo de interfaz de usuario.
- Implementación e implantación del sistema.

Para llevar a cabo las tareas propuestas y arribar satisfactoriamente al resultado final de la investigación se utilizan los siguientes **métodos científicos**:

1. **Analítico-sintético:** en el análisis de diversas metodologías y documentos relacionados con el objeto de estudio de los cuales se extrajeron los rasgos distintivos más importantes relacionados sobre el desarrollo de la Interfaz de usuario UDDI.
2. **Modelación:** este método posibilitó modelar el funcionamiento de una Interfaz de usuario para UDDI, como parte de una propuesta para dar respuesta a las necesidades del proyecto de Informatización.
3. **Observación:** debido a que se pudo constatar los problemas que existían en la universidad producto de la carencia de una interfaz de usuario de un registro UDDI consultable para los desarrolladores y sistemas, precisamente tras estos problemas es que surge la idea de realizar la investigación.
4. **Entrevista:** para la recopilación de toda la información necesaria para el análisis, diseño e implementación de la aplicación, dígase flujo de información, datos de entrada, datos de salida, características de los procesos, etc.

El presente trabajo se encuentra estructurado en **4** capítulos, los cuales resumen la siguiente información:

Capítulo 1. FUNDAMENTACIÓN TEÓRICA: Descripción del objeto de estudio. Sistemas existentes vinculados al campo de acción. Tendencias y tecnologías actuales seleccionadas para el desarrollo de la aplicación y el por qué de su selección. Se explican los principales conceptos a tratar durante la investigación.

Capítulo 2. MODELO DEL SISTEMA: Describe el negocio a través de un modelo de dominio, y se hace el análisis del sistema a desarrollar. Se definen las funcionalidades del sistema y se describen detalladamente.

Capítulo 3. ANALISIS Y DISEÑO: Plantea los detalles relacionados con el análisis y diseño del sistema que se propone desarrollar. Se utilizan para su modelado los diagramas de clases del análisis y diseño web y de interacción para el diseño web, que se necesitan para el almacenamiento de la información persistente.

Capítulo 4. **IMPLEMENTACIÓN Y PRUEBA:** plantea los detalles relacionados con la implementación del sistema que se propone como una primera versión y las pruebas a realizarse en pos de probar la funcionalidad del mismo.

Capítulo

1

FUNDAMENTACIÓN TEÓRICA

1.1 Introducción.

En el presente capítulo se fundamentan cada una de las técnicas, herramientas, metodologías de desarrollo y software utilizados en los que se apoya la solución propuesta del problema. Se brinda una panorámica actual de las soluciones existentes de la problemática planteada, tanto a nivel nacional como internacional, realizando un análisis crítico y valorativo de las tendencias actuales y el estado del arte del tema abordado en el trabajo. Se describen además las tecnologías actuales de desarrollo utilizadas para la modelación e implementación del sistema sobre las cuales se apoya la propuesta.

1.2 Estado de la interfaz de usuario para UDDI a nivel internacional

UDDI y su estado a nivel internacional es un tema muy reciente. A finales de los años 90 emergen las primeras ideas acerca de UDDI, a raíz de una serie de preguntas que fueron surgiendo, enunciadas principalmente sobre como manejar y administrar de manera eficiente y dinámica la creciente y gran cantidad de software que se basan en los servicios web existentes, teniendo en cuenta que existe una colección de software de miles de servicios web, se plantean varias cuestiones difíciles, formuladas con las siguientes preguntas:

- ¿Por qué UDDI? ¿Por qué resulta necesario un registro de este tipo?
- ¿Cómo se descubren los servicios web?
- ¿Cómo se categoriza la información de forma coherente?
- ¿Cómo repercute esto en la localización?
- ¿Cómo afecta a las tecnologías de propietario? ¿Cómo se puede garantizar la interoperabilidad del mecanismo de descubrimiento?

- ¿Cómo se puede interactuar en tiempo de ejecución con este mecanismo de descubrimiento cuando mi aplicación depende de un servicio web?

La iniciativa UDDI surgió como respuesta a estas preguntas. Varias empresas, incluidas Microsoft, IBM, Sun, Oracle, Compaq, Hewlett Packard, Intel, SAP entre otras, unieron sus esfuerzos para desarrollar una especificación basada en estándares abiertos y tecnologías no propietarias que permitiera resolver los retos anteriores. El resultado, cuya versión beta se lanzó en diciembre de 2000 y estaba en producción en mayo de 2001, fue un registro empresarial global alojado por varios nodos de operadores en el que los usuarios podían realizar búsquedas y publicaciones sin coste alguno.
El objetivo de UDDI es facilitar el descubrimiento de servicios tanto dinámicamente como en tiempo de diseño por lo cual grandes compañías como Microsoft, IBM, SAP y NTT Communications ya cuentan con su propia UDDI y cada una de estás con una interfaz gráfica que les permita a los usuarios realizar búsquedas y publicaciones sin coste alguno, con el objetivo de que estos puedan seleccionar con criterios como, calidad de servicio o facilidad de uso de la interfaz web.

1.3 Estado de la interfaz de usuario para UDDI a nivel nacional

Se puede decir que el estado de las UDDI a nivel nacional, es prácticamente cero. En Cuba, existen varios factores los cuales han influido en demorar la llegada de esta tecnología, pero cabe destacar que a partir de los primeros pasos de los desarrolladores cubanos en el campo de la tecnología de los servicios web, se comienza a valorar y pensar en cómo tener un buen control de estos servicios, por tal motivo se tiene en cuenta la tecnología UDDI, como eficiente herramienta que posibilita la administración de los servicios web.
Referente al estado de esta tecnología a nivel nacional, vale señalar que Cuba no cuenta con ninguna organización científica, ni con un grupo de trabajo que pertenezcan a la Sociedad UDDI, en caso contrario, si se contara con ello, esto facilitaría el desarrollo de esta tecnología a nivel nacional, como tampoco en el país se ha realizado una investigación profunda y abarcadora sobre esta nueva tecnología.
Este trabajo está destinado precisamente a eso, se basa en modelar e implementar una interfaz de usuario para una UDDI que sea capaz de gestionar toda la información referente a los servicios web, así como darlos a conocer, teniendo en cuenta las especificaciones internacionales de UDDI como protocolo estándar en la SOA. Con la presente propuesta se quiere que las empresas cubanas, principalmente la UCI, puedan agrupar sus servicios web en un registro, proporcionando un paso más hacia la interoperabilidad entre los sistemas, tema primordial hoy en día para el desarrollo de las aplicaciones.

1.3.1 Estado de la interfaz de usuario para UDDI en la UCI.

El estado de UDDI en la UCI es prácticamente igual al del país a diferencia de los estudios que se han hecho para la elaboración de la misma y la puesta en marcha por un grupo de desarrolladores que permitirán contar a la institución y al mismo en general, con un producto que cumpla las especificaciones y cuente con una interfaz web bien definida, que posibilite a los usuarios contar con una herramienta poderosa y fácil de usar.

1.4 Tendencia y tecnologías actuales.

Cuando surge una nueva tendencia, por lo general es difícil tratar de explicarla, y en especial tratar de que quienes reciban la explicación, perciban la importancia de las implicaciones que tiene para sus intereses. Así las cosas, se tratará de explicar las tecnologías actuales que se utilizan, las que sirven para facilitar el manejo y gestión de los servicios web y las recomendaciones para aplicarlas en los proyectos de tecnología informática.

La tecnología UDDI es una nueva y poderosa herramienta que contribuye al desarrollo de los sistemas distribuidos y está encaminada a proporcionar una plataforma universal para la integración de las transacciones realizadas mediante servicios web, por eso es tan importante que esta tecnología tenga una interfaz de usuario.

Una tendencia es una dirección hacia la cual están trabajando proveedores y consumidores de tecnología informática. Las tecnologías y protocolos que se explicarán a continuación, son simplemente eso, una tendencia, sobre la cual se debe reflexionar y saber que hacia allá va la tecnología y que cada vez se hace más difícil desarrollar aplicaciones en casa que puedan manejar todas estas tendencias, a menos que se seleccionen los estándares adecuados.

1.4.1 Servicios Web.

Un servicio web no es más que un tipo de interfaz de un sistema informático, concretamente se trata de una interfaz que ofrece las funcionalidades del mismo en forma de servicios accesibles a través de los estándares abiertos utilizados en Internet como el protocolo de transferencia de hipertexto (HTTP, HyperText Transfer Protocol). HTTP es el protocolo de aplicación estándar de acceso y transferencia de la información de la web. Las organizaciones de estandarización de estos protocolos son World Wide Web Consortium (W3C), que elevan a la categoría de estándares las especificaciones mencionadas aproximadamente en el año 2000. [1]

Los servicios web llevan la programación orientada a objetos al siguiente nivel de abstracción, pudiendo verse como objetos universales o componentes software que pueden ejecutarse en cualquier sistema operativo. Son realmente código reutilizable. En programación orientada a objetos clásica, los

objetos sólo eran razonablemente reutilizables desde el mismo lenguaje de programación o lenguajes afines, siempre que definieran un protocolo de transporte de mensajes común. [1]

A diferencia de los modelos Cliente/Servidor, tales como un servidor de páginas web, los servicios web no proveen al usuario una interfaz gráfica (GUI). Estos comparten la lógica del negocio**,** los datos y los procesos, por medio de una interfaz de programas a través de la red. Es decir conectan programas, por tanto son programas que no interactúan directamente con los usuarios. Los desarrolladores pueden por consiguiente agregar a los servicios web la interfaz para usuarios, por ejemplo mediante una página web o un programa ejecutable, para entregarles a los usuarios una funcionalidad específica que provee un determinado servicio web. [1]

El concepto de servicios web es el comienzo de una nueva arquitectura para construir sistemas orientados a los servicios. El cambio de un sistema orientado a objetos a uno orientado a servicios es una idea que evolucionó de la nueva perspectiva planteada por la red global de Internet. Los servicios web son un nuevo paradigma en el desarrollo de sistemas distribuidos que proveerá una plataforma para todas las transacciones de negocio a negocio (B2B) en la Internet. [2]

Los servicios web son una colección de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Distintas aplicaciones de software desarrolladas en diferentes lenguajes de programación y ejecutadas sobre cualquier plataforma pueden utilizar los servicios web para intercambiar datos en redes de ordenadores como Internet. Para mejorar la interoperabilidad entre distintas implementaciones de servicios web se ha creado el organismo WS-I, encargado de desarrollar diversos perfiles para definir de manera más exhaustiva estos estándares [2].

Los servicios web son la revolución informática de la nueva generación de aplicaciones que trabajan colaborativamente, en las cuales el software esta distribuido en diferentes servidores.

La industria de la tecnología de la información ha utilizado los servicios web durante más de tres años. En numerosos documentos se describen las ventajas técnicas y empresariales que aportan los servicios web. Un gran número de compañías hace uso de estos servicios en sus entornos de producción. Estos escenarios de clientes demuestran que se han conseguido aplicar en la práctica los objetivos de los servicios web. [1]

Los clientes, analistas del sector industrial, proveedores de sistemas así como las publicaciones periódicas del sector comercial coinciden en identificar un área clave que aún no se ha tratado: la entrega confiable de mensajes. Esta se considera un aspecto crucial para que los servicios web se conviertan en la infraestructura principal de la interconexión heterogénea de los procesos, sistemas y productos de las empresas.

Un sistema de mensajería confiable resulta vital para los servicios web. No es posible resolver muchos problemas empresariales si los participantes no pueden estar seguros de que se realizarán los

intercambios de mensajes. Sin un estándar de los servicios web que proporcione una entrega de mensajes confiable, las aplicaciones implementarán la función necesaria en su lógica empresarial. Este requisito representa una carga para los desarrolladores de lógica empresarial, pero lo que es aún más importante: impide la interoperabilidad debido a que se aportan soluciones incoherentes y diferentes a un problema común. [1]

Por último, un estándar de entrega de mensajes confiable mejorará la eficacia de otros estándares de servicios web como, por ejemplo, la seguridad, las transacciones y los procesos empresariales. Estas mejoras únicamente podrán producirse si la entrega de mensajes confiable es un estándar, en lugar de estar incrustada en la lógica empresarial. La entrega confiable de mensajes garantiza que la arquitectura, los protocolos y las interfaces de los servicios web ofrecerán soluciones seguras, interoperables, transaccionales y sólidas. [1]

1.4.1.1 Ventajas de los servicios Web.

Los servicios web además de lo visto anteriormente, también permiten la comunicación entre aplicaciones o componentes de aplicaciones de forma estándar a través de protocolos comunes (como http) y de manera independiente al lenguaje de programación, plataforma de implantación, formato de presentación o sistema operativo. Un servicio web es un contenedor que encapsula funciones específicas y hace que estas funciones puedan ser utilizadas en otros servidores. Algunas ventajas que presentan los servicios web son: [1]

- Son programables.
- Están basados en XML, que es un lenguaje abierto.
- Son auto-descriptivos.
- Pueden buscar registros de otros servicios web.

Además de estás, los servicios web también presentan ventajas en cuanto a que: [1]

- Aportan interoperabilidad entre aplicaciones de software independientemente de sus propiedades o de las plataformas sobre las que se instalen.

- Fomentan los estándares y protocolos basados en texto, que hacen más fácil acceder a su contenido y entender su funcionamiento.
- Al apoyarse en HTTP, los servicios web pueden aprovecharse de los sistemas de seguridad firewall sin necesidad de cambiar las reglas de filtrado.

- Permiten que servicios y software de diferentes compañías ubicadas en diferentes lugares geográficos puedan ser combinados fácilmente para proveer servicios integrados.
- Permiten la interoperabilidad entre plataformas de distintos fabricantes por medio de protocolos estándar y abiertos.

1.4.1.2 Razones para usar los servicios web.

La principal razón para usar servicios web es que se basan en http sobre TCP (Transmission Control Protocol) en el puerto 80. Dado que las organizaciones protegen sus redes mediante firewalls que filtran y bloquean gran parte del tráfico de Internet, cierran casi todos los puertos TCP salvo el 80, que es, precisamente, el que usan los navegadores. Los servicios web se vehiculan por este puerto, por la simple razón de que no resultan bloqueados. [1]

Otra razón por la que los servicios web son muy prácticos es que pueden aportar gran independencia entre la aplicación que usa el servicio web y el propio servicio. De esta forma, los cambios a lo largo del tiempo en uno no deben afectar al otro. Esta flexibilidad será cada vez más importante, dado que la tendencia a construir grandes aplicaciones a partir de componentes distribuidos más pequeños es cada día más acusada. [1]

1.4.2 Las aplicaciones Web.

Las aplicaciones web son una especialización y concreción de las aplicaciones cliente-servidor, o sea, su arquitectura general es la de un sistema cliente/servidor, donde tanto el cliente (el navegador) como el servidor (el servidor web), y el protocolo mediante el que se comunican (HTTP) son estándar, y no han de ser creados por el desarrollador.

La parte del cliente de las aplicaciones web está formada por el código HTML (HyperText Markup Language) que forma la página web, con opción a código ejecutable mediante los lenguajes script de los navegadores (JavaScript, VBScript, PerlScript) o mediante pequeños programas (applets) en Java. La parte del servidor está formada por un programa o script que es ejecutado por el servidor web, y cuya salida se envía al navegador del cliente. [3]

La creciente popularidad de las aplicaciones web se debe a sus múltiples ventajas, entre las cuales podemos citar: [3]

- **Multiplataforma:** Con un solo programa, un único ejecutable, nuestras aplicaciones pueden ser utilizadas a través de múltiples plataformas, tanto de hardware como de software.

- **Actualización instantánea:** Debido que todos los usuarios de la aplicación hacen uso de un solo programa que radica en el servidor, los usuarios siempre utilizarán la versión más actualizada del sistema.

- **Suave curva de aprendizaje:** Los usuarios, como utilizan la aplicación a través de un navegador, hacen uso del sistema tal como si estuvieran navegando por Internet, por lo cual su acceso es más intuitivo.

- **Fácil de integrar con otros sistemas:** Debido a que se basa en protocolos estándares, la información manejada por el sistema puede ser accedida con mayor facilidad por otros sistemas.

- **Acceso móvil:** El usuario puede acceder a la aplicación con la única restricción de que cuente con un acceso a la red privada de la organización o a Internet, dependiendo de las políticas de dicha organización; puede hacerlo desde una computadora de escritorio, una laptop o desde una agenda electrónica; desde su oficina, hogar u otra parte del mundo.

El desarrollo de aplicaciones web está siendo utilizado en muchas organizaciones, esta situación va ir creciendo indefinidamente. Es por ello que día a día se requieran más programadores capacitados para desarrollos basados en el World Wide Web (WWW). La solución al problema planteado se basa precisamente en la confección de una aplicación web que permita a los usuarios buscar e insertar información acerca de los servicios web.

1.4.3 Estándar HTTP.

HTTP es el protocolo usado en cada transacción de la Web (WWW). HTTP define la sintaxis y la semántica que utilizan los elementos software de la arquitectura web (clientes, servidores, proxies) para comunicarse. Es un protocolo orientado a transacciones y sigue el esquema petición-respuesta entre un cliente y un servidor. Al cliente que efectúa la petición (un navegador o un spider) se lo conoce como "user agent" (agente del usuario). A la información transmitida se la llama recurso y se la identifica mediante una URL. Los recursos pueden ser archivos, el resultado de la ejecución de un programa, una consulta a una base de datos, la traducción automática de un documento.

Una transacción HTTP consiste de un encabezado seguido, opcionalmente, por una línea en blanco y algún dato. El encabezado especificará cosas como la acción requerida del servidor, o el tipo de dato retornado, o el código de estado.

El uso de campos de encabezados enviados en las transacciones HTTP da gran flexibilidad al protocolo. Estos campos permiten que se envíe información descriptiva en la transacción, permitiendo así la autenticación, cifrado e identificación de usuario.
Un encabezado es un bloque de datos que precede a la información propiamente dicha, por lo que muchas veces se hace referencia a él como metadato -porque tiene datos sobre los datos-.
Si se reciben líneas de encabezado del cliente, el servidor las coloca en las variables de ambiente con el prefijo HTTP_ seguido del nombre del encabezado. Cualquier carácter guión (-) del nombre del encabezado se convierte a caracteres "_".

1.4.4 Sistemas Manipuladores de Contenido.

También conocidos como sistemas de gestión de contenidos (CMS) es un software que se utiliza principalmente para facilitar la gestión de webs, ya sea en Internet o en una intranet, y por eso también son conocidos como gestores de contenido web (*Web Content Management* o WCM) [4].

Un CMS consiste en una interfaz que controla una o varias bases de datos donde se encuentra el contenido del sitio. El sistema permite manejar de manera independiente el contenido y el diseño. Así, es posible manipular 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 que permite que estos contenidos sean visibles a todo el público.

Algunos de los sistemas de gestión de contenido con los que se trabaja en nuestra universidad son los siguientes: PHP-Nuke, Post-Nuke, Mambo, Joomla, Plone, Xoops, Drupal, Geeklog.

1.4.4.1 Categorías de los CMS.

Existen distintos tipos de CMS: dedicados a portales web, a foros, a galerías, publicaciones digitales. [5]

La multitud de diferentes CMS existentes se pueden agrupar en las siguientes categorías:

- **Genéricos:** ofrecen la plataforma necesaria para desarrollar e implementar aplicaciones que den solución a necesidades específicas. Pueden servir para construir soluciones de gestión de contenidos, para soluciones de comercio electrónico, blogs, portales. Ejemplos: Zope, OpenCMS, Typo3, Apache lenya.

- **Foros**: sitios que permiten la discusión en línea, donde los usuarios pueden reunirse y discutir temas en los que están interesados. Ejemplos: phpBB, SMF, MyBB, Vanilla.

- **Blogs**: publicación de noticias o artículos en orden cronológico con espacio para comentarios y discusión. Ejemplos: Wordpress, Typo, pMachine Pro, Serendipity.

- **Wikis**: sitio web dónde todos los usuarios pueden colaborar en los artículos, aportando información o reescribiéndola. También permite espacio para discusiones. Ejemplos: Mediawiki, Tikiwiki.

- **eCommerce**: son sitios web para comercio electrónico. Ejemplo: osCommerce

- **Portal**: sitio web con contenido y funcionalidad diversa que sirve como fuente de información o como soporte a una comunidad. Ejemplos: PHPNuke, Postnuke, Joomla, Drupal, Plone, IcyPhoenix, E107, Gekko, Jaws.

- **Galería**: permite administrar y generar automáticamente un portal o sitio web que muestra contenido audiovisual, normalmente imágenes. Ejemplos: Gallery, coppermine, FileBrowser.

- **e-Learning:** sirve para la enseñanza de conocimientos. Los usuarios son los profesores y estudiantes, tenemos aulas virtuales donde se ponen a disposición el material del curso. La publicación de un contenido por un profesor es la puesta a disposición de los estudiantes, en un aula virtual, de ese contenido. Ejemplo: Moodle.

- **Publicaciones digitales:** son plataformas especialmente diseñadas teniendo en cuenta las necesidades de las publicaciones digitales, tales como periódicos, revistas, etc. Ejemplo: ePrints.

1.4.4.2 CMS Drupal.

Drupal es un sistema de gestión de contenido modular y muy configurable. Es un programa de código abierto, con licencia GNU/GPL, escrito en PHP, desarrollado y mantenido por una activa comunidad de usuarios. Destaca por la calidad de su código y de las páginas generadas, el respeto de los estándares de la web, y un énfasis especial en la usabilidad y consistencia de todo el sistema. [6]

El diseño de Drupal es especialmente idóneo para construir y gestionar comunidades en Internet. No obstante, su flexibilidad y adaptabilidad, así como la gran cantidad de módulos adicionales disponibles, hace que sea adecuado para realizar muchos tipos diferentes de sitio web.

Drupal es un sistema de administración de contenido para sitios web. Permite publicar artículos, imágenes, u otros archivos y servicios añadidos como foros, encuestas, votaciones, blogs y administración de usuarios y permisos. Drupal es un sistema dinámico: en lugar de almacenar sus contenidos en archivos estáticos en el sistema de ficheros del servidor de forma fija, el contenido textual de las páginas y otras configuraciones son almacenados en una base de datos y se editan utilizando un entorno web incluido en el producto. [6]
Drupal puede ser instalado en cualquier sistema operativo que soporte un servidor web con librerías PHP y puede trabajar contra cualquier base de datos, ya que utiliza una interfaz de abstracción muy potente para programar independientemente de la tecnología utilizada.

El núcleo de Drupal se complementa con un importante número de extensiones que son las que ofrecen verdadera funcionalidad a este entorno. Podría verse como un enorme juego en el que los desarrolladores deciden cómo combinar y configurar los diferentes módulos para crear su nuevo sitio web.

Este sistema de extensiones hace uso de lo que se conoce como Inversion Control Design Pattern en el que la funcionalidad modular es llamada por el framework sólo en el momento de necesitarla. Este patrón de diseño es muy usado para hacer tests unitarios en orientación a objetos y Drupal lo integra gracias a los llamados hooks (eventos internos o callbacks). [6]

Drupal puede ser utilizado para el prototipado funcional de sitios web o para el testeo o desarrollo final de estos. Una muestra de ello es la organización del sistema de ficheros que permite a los desarrolladores de un proyecto trabajar de forma independiente en cualquiera de sus fases. [6]

La unidad básica de información de Drupal se llama nodo. Un nodo tiene ciertos campos asociados, como una fecha de creación y actualización, un título, un identificador, un cuerpo y otras relaciones externas como el usuario que lo creó o los comentarios (si los hubiera) relacionados. Los nodos son esenciales para la creación de contenido y gracias al Content Construction Kit pueden ser extendidos y modificados. [6]

En Drupal existe también un sistema de menús personalizable desde el cual se puede construir la navegación primaria o secundaria del sitio. Se pueden añadir jerarquías de enlaces internos y externos y automáticamente podrán ser incluidos como bloques en cualquiera de las áreas definidas por el template o plantilla visual. Este sistema de navegación complementado con el sistema de taxonomías de Drupal forman el combinando ideal para el diseño de una navegación coherente y flexible dentro del sitio web. [6]

La organización de los usuarios en roles que definen perfiles o conjuntos de permisos de acceso y acciones es otra de las características fundamentales para la gestión de sitios web con Drupal.

Además se debe mencionar el sistema de temas. Si bien es cierto que dentro de cualquier módulo puede generarse contenido HTML, los estándares establecidos para desarrolladores motivan al uso de funciones con prefijo theme_ para definir la forma en que se mostrará el contenido. De este modo cualquiera de estas funciones podrá ser sobrescrita por el diseñador en su template.

Por supuesto, no son estas las únicas características de la arquitectura del CMS. Entre otras que no se han mencionado cabe destacar el sistema de búsqueda basado en índices sobre nodos y taxonomías, el sistema de generación de formularios o el de gestión de ficheros.

Si bien las características básicas de Drupal son suficientes como para lanzar un sitio completo, el núcleo incluye: administración de usuarios, páginas, vocabulario, comentarios y sindicación. Su potencialidad reside en su estructura modular que permite ser extendida y modificada a gusto y placer del administrador para distintas funciones por medio de módulos de fácil instalación. Entonces este multifacético CMS nos permitiría armar desde un blog personal hasta una intranet corporativa, pasando por bibliotecas digitales, wikis, sitios de e-commerce, álbumes de fotos, etc.

Algunas de las características que presenta Drupal son [6]:
Ayuda on-line: Un robusto sistema de ayuda online y páginas de ayuda para los módulos del 'núcleo', tanto para usuarios como para administradores.

Búsqueda: Todo el contenido en Drupal es totalmente indexado en tiempo real y se puede consultar en cualquier momento.

Código abierto: El código fuente de Drupal está libremente disponible bajo los términos de la licencia GNU/GPL. Al contrario que otros sistemas de 'blogs' o de gestión de contenido propietarios, es posible extender o adaptar Drupal según las necesidades.

Módulos: La comunidad de Drupal ha contribuido con muchos módulos que proporcionan funcionalidades como 'página de categorías', autenticación mediante jabber, mensajes privados, bookmarks, etc.

Personalización*:* Un robusto entorno de personalización está implementado en el núcleo de Drupal. Tanto el contenido como la presentación pueden ser individualizados de acuerdo a las preferencias definidas por el usuario.

URLs amigables: Drupal usa el mod_rewrite de Apache para crear URLs que son manejables por los usuarios y los motores de búsqueda.

Multiplataforma: Drupal ha sido concebido para ser una herramienta multiplataforma. Funciona tanto con Apache como con Microsoft IIS como servidores web sino que también puede utilizarse con diversos Sistemas Operativos como Microsoft Windows, Linux BSD, Solaris o Mac OS X.

Multilenguaje: De la misma forma, Drupal está orientado a un público multilingüe y por ello los contenidos pueden ser fácilmente traducibles, utilizando una interfaz gráfica, integrando herramientas de traducción o bien importando contenidos ya traducidos.

Independencia de la base de datos: La base de datos más utilizada durante la instalación de Drupal es MySQL, pero permite incorporar soporte para otras bases de datos.

1.4.4.2.1 Ventajas de Drupal.

Drupal es un CMS que ofrece varias ventajas al usuario que lo desee utilizar como son:

- Drupal es algo más que un CMS, es también un robusto framework sobre el que se puede desarrollar un sitio web muy complejo y personalizado.
- Drupal facilita la actualización de contenidos, ya que no es necesario disponer de un servidor para llevar a cabo los cambios que se requieran.
- Se convierte de esta manera en una herramienta muy usable tanto para redactores como para usuarios.
- Al permitir la generación de URI's significativas que contienen términos relevantes, se obtiene un buen posicionamiento orgánico del sitio web en los motores de búsqueda.

- Todo el contenido almacenado en Drupal queda organizado en base a categorías, dando como resultado un sitio web limpio y altamente navegable para los usuarios.
- Es una herramienta altamente adaptable a cualquier necesidad, lo que la convierte en apta tanto para sitios web de carácter personal como para grandes portales corporativos. Al ser modulable y personalizable, las posibilidades de crecimiento que ofrece son innumerables.
- Al contar con el respaldo de una extensa y activa comunidad de desarrolladores, se aseguran las actualizaciones continuas en el tiempo.

Drupal es un potente framework de desarrollo, altamente escalable y que permite una fácil adaptación dentro de las metodologías de la ingeniería web. Desde el punto de vista del desarrollador, plantear proyectos con esta herramienta favorece la agilización del proceso de puesta en marcha e implementación, permitiendo dedicar mayor tiempo al análisis y evaluación del sitio.

1.4.5 Modelo Cliente Servidor.

Se dice que la arquitectura Cliente/Servidor es la integración distribuida de un sistema en red, con los recursos, medios y aplicaciones que, definidos modularmente en los servidores, administran, ejecutan y atienden las solicitudes de los clientes; todos interrelacionados física y lógicamente, compartiendo datos, procesos e información. Se establece así un enlace de comunicación transparente entre los elementos que conforman la estructura. Entre las principales características de la arquitectura Cliente/Servidor, se pueden destacar las siguientes: [3]

- El servidor presenta a todos sus clientes una interfaz única y bien definida.
- El cliente no necesita conocer la lógica del servidor, sólo su interfaz externa.
- El cliente no depende de la ubicación física del servidor, ni del tipo de equipo físico en el que se encuentra, ni de su sistema operativo.
- Los cambios en el servidor implican pocos o ningún cambio en el cliente.

Ventajas de la arquitectura cliente-servidor:

- El servidor no necesita potencia de procesamiento, parte del proceso se reparte con los clientes.
- Se reduce el tráfico de red considerablemente. Idealmente, el cliente se conecta al servidor cuando es estrictamente necesario, obtiene los datos que necesita y cierra la conexión dejando la red libre.

Las arquitecturas de dos capas contienen tres componentes distribuidos en dos capas: cliente (solicitante de servicios) y servidor (proveedor de servicios). *Ver Anexo 1*.

Los tres componentes son:

- **Interfaz de usuario al sistema.** Tales como una sesión, entradas de texto, desplegado de menús, etc.
- **Administración de procesamiento.** Tales como la ejecución de procesos, el monitoreado de los mismos y servicios de procesamiento de recursos.
- **Administración de bases de datos.** Tales como los servicios de acceso a datos y archivos.

La arquitectura de software de tres capas emergió en la década de los noventas para solventar las limitaciones de la arquitectura de dos capas. La tercera capa (capa de servicios) se localiza entre la interfaz de usuarios (cliente) y el administrador de datos (servidor). Esta capa intermedia provee de servicios para la administración de procesos (tal como desarrollo, monitoreo y alimentación de procesos) que son compartidos por múltiples aplicaciones. [3]. *Ver Anexo 2*.
El servidor de la capa intermedia (también conocido como servidor de aplicaciones) centraliza la lógica de las aplicaciones, haciendo que la administración de cambios sea más sencilla. En arquitecturas más simples, cualquier cambio en la lógica, implica reescribir todas las aplicaciones que dependan de ésta.

1.4.6 Servidor Web Apache.

Se decidió utilizar el servidor Apache por ser un software de código abierto que funciona sobre cualquier plataforma. Tiene capacidad para servir páginas tanto de contenido estático, como de contenido dinámico a través de otras herramientas soportadas que facilitan la actualización de los contenidos mediante bases de datos, ficheros u otras fuentes de información, es muy potente y altamente configurable. [2]
El servidor Apache es un software que esta estructurado en módulos, es decir, está dividido en muchas porciones de código que hacen referencia a diferentes aspectos o funcionalidades del servidor web. Esta modularidad es intencionada ya que la configuración de cada módulo se hace mediante la configuración de las directivas que están contenidas dentro del módulo. Los módulos del Apache se pueden clasificar en tres categorías: [2]

- **Módulos Base:** Módulo con las funciones básicas del Apache.
- **Módulos Multiproceso**: Son los responsables de la unión con los puertos de la máquina, aceptando las peticiones y enviando a los hijos a atender a las peticiones.
- **Módulos Adicionales:** Cualquier otro módulo que le añada una funcionalidad al servidor.

Las funcionalidades más elementales se encuentran en el módulo base, siendo necesario un módulo multiproceso para manejar las peticiones. Se han diseñado varios módulos multiprocesos para cada uno de los sistemas operativos sobre los que se ejecuta el Apache, optimizando el rendimiento y rapidez del código. [2]

El resto de funcionalidades del servidor se consigue por medio de módulos adicionales que se pueden cargar. Para añadir un conjunto de utilidades al servidor, simplemente hay que añadirle un módulo, de forma que no es necesario volver a instalar el software.

1.4.7 Patrones de Diseño.

Un patrón es un modelo a seguir para realizar algo. Los patrones surgen de la experiencia de seres humanos al tratar de lograr ciertos objetivos, además capturan la experiencia existente y probada para promover buenas prácticas.

Los Patrones de Diseño (Design Patterns) son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces.

Estos se dividen en tres grandes categorías: [2]

- **Patrones Creacionales:** solucionan problemas de creación de instancias. Nos ayudan a encapsular y abstraer dicha creación.
- **Patrones Estructurales:** solucionan problemas de composición (agregación) de clases y objetos.
- **Patrones de Comportamiento:** soluciones respecto a la interacción y responsabilidades entre clases y objetos, así como los algoritmos que encapsulan.

Un patrón de diseño es: [3]

- Una solución estándar para un problema común de programación.
- Una técnica para flexibilizar el código haciéndolo satisfacer ciertos criterios.
- Un proyecto o estructura de implementación que logra una finalidad determinada.
- Un lenguaje de programación de alto nivel.
- Una manera más práctica de describir ciertos aspectos de la organización de un programa.
- Conexiones entre componentes de programas.
- La forma de un diagrama de objeto o de un modelo de objeto.

Características:

- Son soluciones concretas.
- Son soluciones técnicas.

- Se utilizan en situaciones frecuentes.
- Favorecen la reutilización de código.
- El uso de un patrón no se refleja en el código.
- Es difícil reutilizar la implementación de un patrón.

Los patrones de diseño han contribuido a dar flexibilidad y extensibilidad a los diseños. Pero en adición, han demostrado ser una forma muy útil (exitosa) de reutilizar diseño, ya que ellos no sólo nombran, abstraen e identifican aspectos claves de estructuras comunes de diseño, sino que generalmente son descritos en una forma específica documental, haciendo su comprensión y aplicación fácil para el conjunto de desarrolladores. [3]

Podemos decir que los beneficios que un patrón produce pueden ser medidos en varios sentidos:

- Contribuyen a reutilizar diseño, identificando aspectos claves de la estructura de un diseño que puede ser aplicado en una gran cantidad de situaciones. La importancia de la reutilización del diseño no es despreciable, ya que ésta nos provee de numerosas ventajas: reduce los esfuerzos de desarrollo y mantenimiento, mejora la seguridad, eficiencia y consistencia de nuestros diseños, y nos proporciona un considerable ahorro en la inversión.
- Mejoran (aumentan, elevan) la flexibilidad, modularidad y extensibilidad, factores internos e íntimamente relacionados con la calidad percibida por el usuario. [7]
- Incrementan nuestro vocabulario de diseño, ayudándonos a diseñar desde un mayor nivel de abstracción.

Ventajas de los patrones de diseño:

- Los patrones de diseño proponen una forma de reutilizar la experiencia de los desarrolladores, para ello clasifica y describe formas de solucionar problemas que ocurren de forma frecuente en el desarrollo.
- Están basados en la recopilación del conocimiento de los expertos en desarrollo de software.
- Es una experiencia real, probada y que funciona. Es Historia y nos ayuda a no cometer los mismos errores.

Por lo tanto la dirección de informatización de la UCI estableció el uso del patrón Modelo Vista Controlador en la realización del sistema.

1.4.8 Modelo Vista Controlador (MVC).

Modelo Vista Controlador (MVC) es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.

El patrón MVC se ve frecuentemente en aplicaciones web, donde la vista es la página HTML y el código que provee de datos dinámicos a la página. [3]

Son muchas las empresas que deciden pasar sus aplicaciones a la arquitectura modelo-vista-controlador para documentar más fácilmente el código, ahorrar espacio y en caso de no disponer de diseñadores web, poder contratar los servicios de un diseñador que no sepa mucho de programación que les haga las vistas.

El **Modelo** es todo acceso a datos, y las funciones que llevan se llaman "lógica de negocio", o sea datos y reglas de negocio. Lleva un registro de las vistas y controladores del sistema. Cada acceso a datos se pone en su función individual porque, de esta forma, si se cambia de gestor de bases de datos este cambio sólo afecta a estas funciones, no al resto de la aplicación. Tener el modelo bien delimitado permite la existencia de varias aplicaciones que compartan el mismo modelo. [2]

La **Vista**, en una aplicación web, es el HTML y lo necesario para convertir datos en HTML, o sea muestra la información del modelo al usuario. Tienen un registro de su controlador asociado (normalmente porque además lo instancia). [2]

Pueden dar el servicio de "Actualización ()", para que sea invocado por el controlador o por el modelo. Tener la vista separada del controlador permite cambiar la aplicación para que genere, en lugar de HTML, algo distinto (por ejemplo, WML), sin tener que tocar más que una parte completamente delimitada del código. [2]

El **Controlador** es lo que une la vista y el modelo. Por ejemplo, son las funciones que toman los valores de un formulario, consultan la base de datos (a través del modelo) y producen valores, que la vista tomará y convertirá en HTML. En resumen, gestiona las entradas del usuario. Recibe los eventos de entrada (un clic, un cambio en un campo de texto, etc.).

Contiene reglas de gestión de eventos, del tipo "SI Evento Z, entonces Acción W". Estas acciones pueden suponer peticiones al modelo o a las vistas. De este modo, el código que "hace algo" está perfectamente separado del código dedicado a crear HTML. [2]

Un ejemplo típico del funcionamiento de este patrón se puede observar en el *Anexo 3*. La estructura de este patrón se puede observar en el *Anexo 4*.

El MVC es un patrón ampliamente utilizado en múltiples plataformas y lenguajes.

Algunos de sus principales beneficios son: [2]

- Menor acoplamiento.

- Desacopla las vistas de los modelos.
- Desacopla los modelos de la forma en que se muestran e ingresan los datos.

- Mayor cohesión.
  - Cada elemento del patrón está altamente especializado en su tarea.
- Las vistas proveen mayor flexibilidad y agilidad.
  - Se puede crear múltiples vistas de un modelo.
  - Se puede crear, añadir, modificar y eliminar nuevas vistas dinámicamente.
  - Las vistas pueden anidarse.
  - Se puede cambiar el modo en que una vista responde al usuario sin cambiar su representación visual.
  - Se puede sincronizar las vistas.
  - Las vistas pueden concentrarse en diferentes aspectos del modelo.
- Mayor facilidad para el desarrollo de clientes ricos en múltiples dispositivos y canales.
  - Una vista para cada dispositivo que puede variar según sus capacidades.
  - Una vista para la web y otra para aplicaciones de escritorio.
- Más claridad de diseño.
- Facilita el mantenimiento.
- Mayor escalabilidad.

Entre las ventajas del estilo Modelo-Vista-Controlador están las siguientes:

- **Soporte de múltiples vistas:** Dado que la vista se halla separada del modelo y no hay dependencia directa del modelo con respecto a la vista, la interfaz de usuario puede mostrar múltiples vistas de los mismos datos simultáneamente. Por ejemplo, múltiples páginas de una aplicación web pueden utilizar el mismo modelo de objetos mostrado de maneras diferentes.

- **Adaptación al cambio:** Los requerimientos de interfaz de usuario tienden a cambiar con mayor rapidez que las reglas de negocios. Los usuarios pueden preferir distintas opciones de representación, o requerir soporte para nuevos dispositivos como teléfonos celulares o PDAs. Dado que el modelo no depende de las vistas, agregar nuevas opciones de presentación generalmente no afecta al modelo.

Una **desventaja** que tiene este modelo es el costo de actualizaciones frecuentes: Si el modelo experimenta cambios frecuentes, por ejemplo, podría desbordar las vistas con una lluvia de requerimientos de actualización.

1.4.9 PHP.

PHP (Hypertext Preprocessor), es un lenguaje interpretado de alto nivel embebido en páginas HTML y ejecutado en el servidor. En mayo del 2000, PHP 4 fue lanzado bajo el poder del motor Zend Engine 1.0. El 13 de julio de 2004, PHP 5 fue lanzado, utilizando el motor Zend Engine II. La versión más reciente de PHP es la 5.1.4, que incluye el novedoso PDO (PHP Data Objects) y mejoras utilizando las ventajas que provee el nuevo Zend Engine II. Según estudios, más de un millón de servidores tienen esta capacidad implementada y los números continúan creciendo.

Una de sus características más potentes es su soporte para gran cantidad de bases de datos. Entre las que se pueden mencionar InterBase, mSQL, MySQL, Oracle, Informix, PostgreSQL, entre otras. PHP también ofrece la integración con varias bibliotecas externas, que dan al desarrollador la posibilidad de realizar cualquier tarea, desde generar documentos en PDF (Portable Document Format) hasta analizar código XML y últimamente también para la creación de otro tipo de programas incluyendo aplicaciones con interfaz gráfica usando la librería GTK+. [3]

Es software libre, lo que implica menos costes y servidores más baratos que otras alternativas. Es muy rápido y su integración con la base de datos MySQL y el servidor Apache, le permite constituirse como una de las alternativas más atractivas del mercado. [2]

Es multiplataforma, funciona tanto para Unix (con Apache) como para Windows (con Microsoft Internet Information Server) de forma que el código que se haya creado para una de ellas no tiene porque modificarse al pasar a la otra.

Su sintaxis está inspirada en C, ligeramente modificada para adaptarlo al entorno en el que trabaja, de modo que si se está familiarizado con esta sintaxis, le resultará muy fácil aprender PHP. Su librería

estándar es realmente amplia, lo que permite reducir los llamados "costes ocultos", uno de los principales defectos de ASP (Active Server Pages).

Debido a su amplia distribución PHP esta perfectamente soportado por una gran comunidad de desarrolladores. Como producto de código abierto, PHP goza de la ayuda de un gran grupo de programadores, permitiendo que los fallos de funcionamiento se encuentren y se reparen rápidamente. El código se pone al día continuamente con mejoras y extensiones de lenguaje para ampliar las capacidades del mismo.

- El funcionamiento del PHP se puede describir a través de los pasos siguientes: [3] Escribir en las páginas HTML pero con el código PHP dentro.
- Guardar la página en el servidor web.
- Un navegador solicita una página al servidor.
- El servidor interpreta el código PHP.
- El servidor envía el resultado del conjunto de código HTML y el resultado del código PHP que también es HTML.

Después de seis años, y después que la comunidad ha revisado el paquete de legados que ha dejado el PHP, se han realizado cambios estructurales en el lenguaje para ofrecer innovación en el nuevo PHP 5 y solucionar muchos de los problemas encontrados en PHP 4.

Afortunadamente, lo nuevo de PHP 5 mejora muchas áreas en el lenguaje y su ejecución, como por ejemplo:

- Programación orientada a objetos (POO).
- MySQL.
- XML.
- Integración nativa con el Zend Engine.

Otros lenguajes como Perl (Practical Extraction and Report Language), ASP y JSP (Java Server Pages) tienen características similares al PHP aunque poseen rasgos que los marcan y por ello los distingue, entre ellas podemos encontrar: [3]

- **Multiplataforma**: Menos el ASP, que es solamente soportado por la plataforma Windows, los demás lenguajes están soportados en múltiples plataformas.
- **Velocidad de ejecución**: la velocidad es mayor en PHP, seguidos por PERL y JSP.
- **Disponibilidad** de recursos: actualmente los más utilizados en la Internet son el PHP y el JSP, siendo más utilizado en la publicación de artículos y códigos de ejemplos. PHP tiene una de las comunidades más grandes en Internet, al igual que la de Java.

- **Familiaridad con el lenguaje**: En la universidad los lenguajes más utilizados por los programadores son el ASP y el PHP.
- **De acuerdo a estas comparaciones**, el PHP resulta mucho más favorecido, por tanto la dirección de informatización de la UCI ha determinado el uso de PHP 5 como herramienta para implementar la propuesta de sistema de este trabajo.

1.4.10 JavaScript.

Javascript es un lenguaje con muchas posibilidades, utilizado para crear pequeños programas que luego son insertados en una página web y en programas más grandes, orientados a objetos mucho más complejos. Con Javascript podemos crear diferentes efectos e interactuar con nuestros usuarios.
Se trata de un lenguaje de programación del lado del cliente, porque es el navegador el que soporta la carga de procesamiento. Gracias a su compatibilidad con la mayoría de los navegadores modernos, es el lenguaje de programación del lado del cliente más utilizado.
Este lenguaje posee varias características, entre ellas podemos mencionar que es un lenguaje basado en acciones que posee menos restricciones. Además, es un lenguaje que utiliza Windows y sistemas X-Windows, gran parte de la programación en este lenguaje está centrada en describir objetos, escribir funciones que respondan a movimientos del mouse, aperturas, utilización de teclas, cargas de páginas entre otros.

JavaScript es un lenguaje de programación interpretado, es decir, que no requiere compilación, que se utiliza principalmente para crear páginas web dinámicas.

Técnicamente, JavaScript es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios.

1.4.11 PostgreSQL.

PostgreSQL es un sistema de gestión de bases de datos Objeto-Relacionales (ORDBMS) libre, bajo la licencia BSD (Berkeley Software Distribution). Es una alternativa a otros sistemas de bases de datos de código abierto (como MySQL, Firebird y MaxDB), así como sistemas propietarios como Oracle y SQLServer.
PostgreSQL está ampliamente considerado como el sistema de bases de datos de código abierto más avanzado del mundo. Posee muchas características que tradicionalmente sólo se podían ver en productos comerciales de alto calibre.
La siguiente es una breve lista de algunas de esas características, a partir de PostgreSQL 7.1.x. [8]

DBMS Objeto-Relacional: PostgreSQL aproxima los datos a un modelo objeto-relacional, y es capaz de manejar complejas rutinas y reglas. Ejemplos de su avanzada funcionalidad son consultas SQL declarativas, control de concurrencia multi-versión, soporte multi-usuario, transacciones, optimización de consultas, herencia, y arreglos.

Altamente Extensible: PostgreSQL soporta operadores, funciones, métodos de acceso y tipos de datos definidos por el usuario.

Soporte SQL Comprensivo: PostgreSQL soporta la especificación SQL99 e incluye características avanzadas tales como las uniones (joins) SQL92.

Integridad Referencial: PostgreSQL soporta integridad referencial, la cual es utilizada para garantizar la validez de los datos de la base de datos.

Lenguajes Procedurales: PostgreSQL tiene soporte para lenguajes procedurales internos, incluyendo un lenguaje nativo denominado PL/pgSQL. Este lenguaje es comparable al lenguaje procedural de Oracle, PL/SQL.

MVCC: MVCC, o Control de Concurrencia Multi-Versión (Multi-Version Concurrency Control), es la tecnología que PostgreSQL usa para evitar bloqueos innecesarios. Los bloqueos son provocados por usuarios que están escribiendo en la base de datos. Resumiendo, el lector está bloqueado por los escritores que están actualizando registros.

Cliente/Servidor: PostgreSQL usa una arquitectura proceso-por-usuario cliente/servidor. Esta es similar al método del Apache 1.3.x para manejar procesos. Hay un proceso maestro que se ramifica para proporcionar conexiones adicionales para cada cliente que intente conectar a PostgreSQL.

Otra ventaja de PostgreSQL es su habilidad para usar Perl, Python, o TCL (Tool Command Language) como lenguaje procedural embebido.

Las principales mejoras en PostgreSQL incluyen: [8]

- Los bloqueos de tabla han sido sustituidos por el control de concurrencia multi-versión, el cual permite a los accesos de sólo lectura continuar leyendo datos consistentes durante la actualización de registros, y permite copias de seguridad en caliente desde pg_dump mientras la base de datos permanece disponible para consultas.
- Se han implementado importantes características del motor de datos, incluyendo sub consultas, valores por defecto, restricciones a valores en los campos (constraints) y disparadores (triggers).
- Se han añadido características adicionales que cumplen el estándar SQL92, incluyendo claves primarias, identificadores entrecomillados, forzado de tipos cadenas literales, conversión de tipos y entrada de enteros binarios y hexadecimales.

- Los tipos internos han sido mejorados, incluyendo nuevos tipos de fecha/hora de rango amplio y soporte para tipos geométricos adicionales.
- La velocidad del código del motor de datos ha sido incrementada aproximadamente en un 20-40%, y su tiempo de arranque ha bajado el 80% desde que la versión 6.0 fue lanzada.

1.4.12 Proceso de Desarrollo.

1.4.12 .1 ¿Qué metodología debo usar para el desarrollo de un Software?

Cada día la producción de software busca adecuarse más a las necesidades del usuario, esto trae como consecuencia que aumente en tamaño y complejidad. Para lograr la productividad del software se necesita un proceso que integre las múltiples facetas del desarrollo del mismo. Se hace necesario definir la metodología de ingeniería del software que guiará el proceso de automatización, se ha escogido el Proceso Unificado de Desarrollo de Software (RUP, Rational Unified Process).

RUP es un conjunto de actividades necesarias para transformar los requisitos de un usuario en un sistema de software. Es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas de software, para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de aptitud y diferentes tamaños de proyectos. [9]

Es un proceso basado en componentes, que utiliza el Lenguaje Unificado de Modelado (UML) para preparar todos los esquemas de un sistema software. No obstante, los verdaderos aspectos definitorios del Proceso Unificado se resumen en que está dirigido por casos de uso, este avanza a través de una serie de flujos de trabajo, que parten de los casos de uso; centrado en la arquitectura y es iterativo e incremental. [13]. *Ver Anexo 5*.

Está acompañado de una herramienta muy buena que soporta cada uno de los procesos que necesitamos: Visual Paradigm 6.0 Enterprise Edition. Además cubre el ciclo de vida de desarrollo de un proyecto y toma en cuenta las mejores prácticas a utilizar en el modelo de desarrollo de software.

Lenguaje Unificado de Modelado (UML).

UML (Unified Modeling Language) es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema de software. Sus creadores pretendieron con este lenguaje, unificar las experiencias acumuladas sobre técnicas de modelado e incorporar las mejores prácticas en un acercamiento estándar. [10]

El UML permite a los creadores de sistemas generar diseños que capturen sus ideas en una forma convencional y fácil de comprender para comunicarlas a otras personas que estén involucradas en el proceso de desarrollo de los sistemas, esto se lleva a cabo mediante un conjunto de símbolos y diagramas.

El UML está compuesto por diversos elementos gráficos que se combinan para conformar diagramas y proporciona un estándar que permite al analista de sistemas generar un anteproyecto de varias facetas que sean comprensibles para los clientes, desarrolladores y todos aquellos que estén involucrados en el proceso de desarrollo.

Un modelo UML indica que es lo que supuestamente hará el sistema pero no como lo hará.

De forma general las principales características son: [10]

- Lenguaje unificado para la modelación de sistemas.
- Tecnología orientada a objetos.
- El cliente participa en todas las etapas del proyecto.
- Corrección de errores viables en todas las etapas.
- Aplicable para tratar asuntos de escala inherente a sistemas complejos de misión crítica, tiempo real y cliente/servidor.

Las ventajas del UML son: [10]

- Diseño y documentación.
- Código reutilizable.
- Descubrimiento de fallas.
- Ahorro de tiempo en el desarrollo del software.
- Mucho más fáciles las modificaciones.
- Más fácil la comunicación entre programadores.

Estándares que conforman UML: [10]

- **Superestructura:** es aquí dónde se definen los diagramas y los elementos que los componen.
- **Infraestructura:** conceptos de bajo nivel. Meta-Modelo, da soporte a la superestructura, entre otras.
- **OCL:** lenguaje de restricción. De utilidad para especificar conceptos ambiguos sobre los distintos elementos del diagrama.
- **XMl / Intercambio de diagramas:** Permite compartir diagramas entre diferentes herramientas de modelado UML.

Existen varias herramientas CASE (Computer-Aided Systems Engineering), que dan asistencia a analistas, ingenieros de software y desarrolladores durante el ciclo de vida de desarrollo de un software y entre los principales objetivos de estas herramientas se encuentran los siguientes:

Objetivos de las herramientas CASE:

- Mejorar la productividad en el desarrollo y mantenimiento del software.
- Aumentar la calidad del software.
- Mejorar el tiempo y coste de desarrollo y mantenimiento de los sistemas informáticos.
- Mejorar la planificación de un proyecto.
- Aumentar la biblioteca de conocimientos informáticos de una empresa ayudando a la búsqueda de soluciones para los requisitos.
- Automatizar el desarrollo del software, la documentación, la generación de código, las pruebas de errores y gestión del proyecto.
- Ayuda a la reutilización del software, portabilidad y estandarización de la documentación.
- Gestión global en todas las fases de desarrollo de software con una misma herramienta.
- Facilitar el uso de las distintas metodologías propias de la ingeniería del software.

1.5 Herramientas utilizadas.

1.5.1 Diseño de interfaz.

1.5.1.1 Nvu (which stands for "new view").

NVU es un editor de páginas Web WYSIWYG (What You See Is What You Get) multiplataforma basado en Mozilla Composer, pero de ejecución independiente. Añade características nuevas como soporte integrado de CSS y mejor gestión del soporte FTP para actualización de los ficheros. [11]

Este editor facilita el desarrollo de páginas web, gracias a las diferentes visualizaciones disponibles en su interfaz (código fuente, ventana WYSIWYG, visión con tags de HTML realzados), entre los cuales es posible cambiar mediante un sistema de pestañas. [11]

Incluye también otras características como gestión de trabajo mediante proyectos, cliente FTP integrado para subir la página directamente desde NVU y soporte para todos los elementos típicos: marcos, formularios, tablas, plantillas de diseño, hojas de estilo CSS, etc. [11]

NVU está disponible para Linux, Mac OS X y Microsoft Windows. [11]

Hay una versión portátil de NVU que puede ser transportada y usada directamente desde una memoria USB sin necesidad de instalarse en el computador. [11]

Por último se trata de una aplicación con licencia GPL (General Public License), que brinda a los usuarios varias libertades básicas, entre ellas la posibilidad de hacer copias del programa. Este hecho es una ventaja (por ejemplo) si estamos en el mundo de la enseñanza, ya que podemos utilizar con nuestros alumnos y alumnas esta aplicación y podemos distribuirla para que la instalen en sus

domicilios (si no tienen conexión a Internet) y puedan trabajar con la misma aplicación que se usa en los centros educativos. [11]

1.5.2 Zend Studio.

Zend Studio es uno de los ambientes de desarrollo integrado o Integrated Development Environment (IDE) disponible para desarrolladores profesionales que agrupa todos los componentes de desarrollo necesarios para el ciclo de desarrollo de aplicaciones en PHP. A través de un comprensivo conjunto de herramientas de edición, depurado, análisis, optimización y bases de datos, Zend Studio acelera los ciclos de desarrollo y simplifica los proyectos complejos.

1.5.3 GIMP.

GIMP (GNU Image Manipulation Program) es un programa de edición de imágenes. Es un programa libre y gratuito, englobado en el proyecto GNU y disponible bajo la licencia GNU.

La primera versión de GIMP se desarrolló para sistemas Unix y fue pensada especialmente para GNU/Linux. Existen versiones totalmente funcionales para Windows, para Mac OS X y para otros sistemas operativos, haciéndolo el programa de manipulación de gráficos disponible en más sistemas operativos. Se le puede considerar como la alternativa más firme para Photoshop, aunque posee una interfaz muy diferente.

Existe una versión portátil de GIMP que puede ser transportada y usada directamente desde una memoria USB sin necesidad de instalarse en el ordenador, disponible solo para ordenadores bajo Windows y Mac.

La biblioteca de controles gráficos GTK, desarrollada para GIMP, dio origen al entorno de ventanas de GNOME.

1.5.4 Visual Paradigm.

Visual Paradigm es una herramienta gratuita utilizada para el modelado de aplicaciones, utiliza UML como lenguaje de modelado, está diseñada para una gran cantidad de usuarios. Esta herramienta visual permite construir la aplicación con mayor rapidez, mayor exactitud y mejor trabajo en equipo permitiendo aumentar las expectativas mediante la interfaz gráfica. Facilita la interoperabilidad con otras herramientas CASE y permite la integración de todos los componentes.

1.6 Conclusiones.

En este capítulo se exponen las condiciones y problemas que rodean el objeto de estudio, a través de los conceptos y definiciones planteadas. Se evidencia la necesidad de implementar un software, en este caso, una interfaz de usuario que permita la interoperabilidad, organización y publicación de los servicios web en el contexto de la arquitectura SOA para la UCI. Para desarrollar el sistema se hace uso de la tecnología para la programación de páginas dinámicas el lenguaje PHP5 y con soporte de base de datos en PostgreSQL. El proceso de desarrollo es RUP, el cual está basado en el lenguaje de modelado UML, que permite incorporar al proceso de desarrollo de software un mejor control de los requerimientos y cambios. Además se justifican las herramientas y tecnologías que son utilizadas, todas compatibles con software libre, así como, sus características y peculiaridades.

Capítulo

2

MODELO DEL SISTEMA

2.1 Introducción.

En este capítulo se realiza un estudio de la situación actual del problema a resolver a través de una descripción de los procesos de negocios, como resultado de la inexistencia del mismo se necesita realizar un Modelo de Dominio, en el cual se definen conceptos que permiten realizar una correcta captura de requisitos y poder construir un sistema correcto.

Además se definen los requerimientos funcionales y no funcionales que debe tener la propuesta del sistema, lo que posibilita que se puedan identificar actores y casos de uso y construir un Diagrama de Casos de Usos del Sistema.

2.2 Descripción de los Proceso del Negocio Propuestos.

Para describir los procesos del negocio que se relacionan con el campo de acción de este trabajo, es necesario conocer qué tipo de topología de UDDI aplicar en la UCI, para centrar la atención en los principales procesos a realizar, siendo necesario definir las características y objetivos de la universidad y así elegir la más conveniente.

La UCI es un centro de estudio, que aborda de manera íntegra el estudio de la informática, en ella se preparan futuros especialistas en esta rama. Tiene como objetivos principales la formación de profesionales en Informática y un segundo objetivo muy importante que es la producción de software para el comercio y la informatización de la sociedad cubana. Para el uso de UDDI existen topologías como:

Registro de Negocios UDDI (UBR): Es una red de registros UDDI públicos en Internet.

E-Mercado UDDI: Registro UDDI dedicado a un sector de la industria dado.

Nodo Privado UDDI: Registro UDDI sobre una intranet, usado para proveer la funcionalidad de UDDI a los usuarios de una compañía.

Nodo Extranet UDDI: Registro UDDI sobre una extranet para proveer la funcionalidad de UDDI entre compañías relacionadas en el negocio.

Como ya se ha dicho, en la UCI se vaticina que en un futuro cercano existan una gran cantidad de servicios web, por lo que se propone se implante una UDDI con una topología de tipo **Nodo Privado UDDI**, este proveerá la funcionalidad de UDDI a través de la Intranet a los usuarios de la UCI, este registro si se desea no será accesible desde Internet ni por ninguna otra institución, por lo que sus datos no se replicarán con ningún otro servidor UDDI. Este tipo de topología trae consigo que se puedan definir las propias políticas de seguridad y acceso.

De esta manera la UCI contará con una excelente herramienta para la publicación, búsqueda y categorización de sus servicios web que serán reutilizados por parte de los desarrolladores de aplicaciones.

Proceso de Administración.

Este proceso tiene como objetivo controlar toda la información existente y administrar las funcionalidades del sistema, con la idea de que el administrador del Nodo Privado UDDI sea el encargado de garantizar que cumpla con todos los requisitos establecidos.

Con la propuesta de este trabajo, se trataría de que todos los usuarios que tengan acceso puedan incorporar al catálogo de servicio, información de un servicio del proveedor del servicio web o de la especificación técnica.

Proceso de Publicación.

Este proceso tiene como objetivo registrar toda la información que se puede necesitar para el desarrollo del sistema, siendo obligatorio que el especialista que pertenezca a la UCI se registre en el Nodo Privado UDDI para convertirse así en un publicador, lo que permitirá que pueda publicar proveedores de servicios web y especificaciones técnicas.

La información se distribuye en 2 partes fundamentales, registro de proveedores y especificaciones técnicas. Dentro del registro de proveedores, se guarda toda la información referente a éstos, ya sea, nombre del mismo, servicios web, contactos, identificadores, categorías, URL y relaciones con servicios de otros proveedores. Por otra parte para acceder a las especificaciones técnicas de un servicio se recurre al registro que contiene información acerca de ellas, además del identificador, URL del documento de la especificación y la categoría.

Proceso de Búsquedas.

Este proceso tiene como objetivo mostrar la información necesaria para desarrollar un sistema informático, con la idea de que cualquier persona que trabaje o estudie en la UCI pueda utilizar el

Nodo Privado UDDI para consultar información de los servicios, proveedores, categorías o especificaciones técnicas de los mismos.

La información se distribuye en 3 partes: para listar los servicios se recurre a la búsqueda de servicio, esta puede ser por el nombre o por las especificaciones técnicas, para listar las especificaciones técnicas se recurre a la búsqueda de especificaciones técnicas por el nombre, y finalmente para listar los proveedores se recurre a la búsqueda del proveedor por el nombre o por las especificaciones técnicas.

A través del sistema, se trataría de agrupar toda información de los servicios, proveedores o especificaciones técnicas del servicio en un catálogo en la UCI, de manera tal que una vez que se solicite una de estas opciones, el sistema busque la información, y siempre que exista, se visualice el contenido.

2.3 Modelo del Dominio.

Teniendo en cuenta las descripciones de los procesos en el epígrafe anterior, se concluye que para dar solución al negocio que se está estudiando es necesario realizar un Modelo de Dominio, debido a la inexistencia de una Interfaz de usuario para UDDI en la universidad. El Modelo del dominio es el encargado de capturar los tipos más importantes de objetos que existen o los eventos que suceden en la UCI, permitiendo de manera visual mostrar al usuario los principales conceptos que se manejan en el dominio del sistema en desarrollo. Esto ayuda a los clientes, desarrolladores e interesados a utilizar un vocabulario común para poder entender el contexto en que se enmarca el sistema. Este modelo va a contribuir posteriormente a identificar algunas clases que se utilizarán. El primer paso para ello es definir los principales conceptos.

2.3.1 Conceptos.

• Se le denominará **usuario,** a aquel usuario que no tiene que registrarse para buscar información acerca de los servicios que están publicados.

• Se le denominará **publicador**, a aquel usuario que tenga permiso o acceso para gestionar (registrar, eliminar, modificar, buscar) uno o varios datos dentro de la UDDI.

• Se le denominará **administrador**, a aquel usuario que tenga permiso o acceso para administrar (gestionar, buscar, administrar usuarios) todos los datos dentro de la UDDI.

• Se le denominara **institución**, a todas las empresas que pertenezcan a la asociación UDDI, en este caso, la UCI.

• Se denominará **registro UDDI**, a todo el sistema que permita registrar, buscar información de forma estructurada sobre los servicios web, proveedores de estos servicios y las especificaciones técnicas de los mismos.

• Se denominará **información UDDI**, a todo término que brinde información acerca de sus servicios web, proveedor del servicio y las especificaciones técnicas de éstos.

2.3 Modelo del Dominio.

DIAGRAMA DEL MODELO DEL DOMINIO

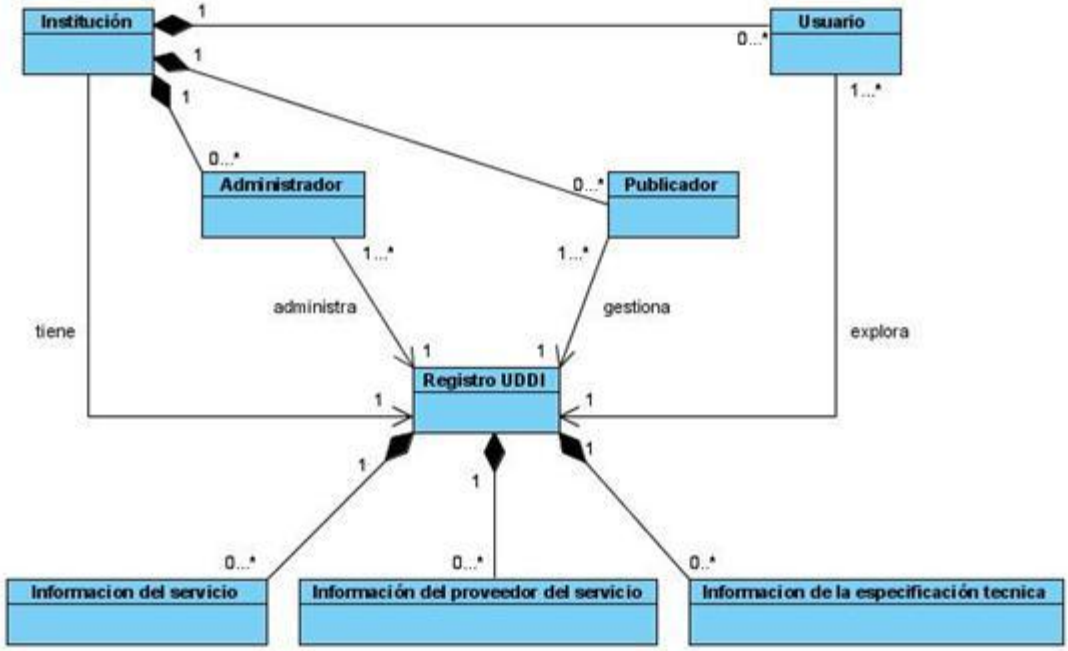


Figura 2.1 Modelo de Dominio.

2.4 Requerimientos.

2.4.1 Requerimientos Funcionales.

Los requisitos funcionales son capacidades o necesidades desde el punto de vista del usuario que debe cumplir el sistema y que están fuertemente ligados a las opciones del software.

Para cumplir con los objetivos propuestos se prevé que el sistema tenga las siguientes funcionalidades:

RF1- Autenticar usuarios.

RF1.1- Proporcionar los permisos necesarios para cada operario de la aplicación según sus roles.

RF2- Buscar información de servicio web.

RF2.1- Buscar especificación técnica. RF2.2- Buscar servicios.

RF2.2.1- Listar servicios por especificación técnica.

RF2.3 - Buscar proveedores.

RF2.3.1 - Listar proveedores por especificación técnica.

RF3- Gestionar la información de servicios web.

RF3.1-Gestionar información de proveedores.

RF3.1.1- Listar proveedores.

RF3.1.2- Adicionar proveedores.

RF3.1.3- Eliminar proveedores.

RF3.2 - Gestionar información de especificación técnica.

RF3.2.1 - Listar especificación técnica.

RF3.2.2 - Adicionar especificación técnica.

RF3.2.3 - Eliminar especificación técnica.

RF4- Administrar sistema.

RF5- Administrar usuarios.

RF5.1 - Registrar usuario.

RF5.2 - Eliminar usuario.

RF5.3 - Modificar usuario.

RF6- Controlar acceso a la información.

RF6.1 – Añadir rol.

RF6.2 – Modificar rol.

RF6.3 – Eliminar rol.

2.4.2 Requerimientos no Funcionales.

Los requerimientos no funcionales son características que describen alguna forma o restricción para la realización de algún requerimiento (funcionalidad), conjunto de ellos o todos. Se consideran los atributos del sistema, propiedades o cualidades que debe tener el producto.

A continuación se muestran los requerimientos no funcionales:

- **Apariencia o interfaz externa:**

La interfaz no contiene muchas imágenes para no demorar las respuestas al usuario. El diseño de la interfaz es sencillo y claro de usar con reconocimiento visual a través de elementos visibles que identifiquen cada una de sus acciones. Es formal, serio y con una navegación sugerente, todo esto teniendo en cuenta el fin con el que se desarrolla la aplicación.

• Usabilidad:

El sistema puede ser usado por cualquier persona que posea conocimientos básicos en el manejo de la computadora y de un ambiente web en sentido general.

• Rendimiento:

La disponibilidad de trabajo en red contra el servidor es constante. Se garantiza que la respuesta a solicitudes de los usuarios del sistema, sea en un período de tiempo breve (de segundos) para evitar la acumulación de trabajo por parte de los responsables. El sistema deberá ser lo más estable y confiable posible.

• Soporte:

Se requiere que el producto reciba mantenimiento ante cualquier fallo que ocurra.

• Portabilidad:

El producto es multiplataforma, corre sobre una plataforma web, codificada en “PHP 5” y su sistema de bases de datos en PostgreSQL.

• Seguridad:

El sistema se encarga de controlar los diferentes niveles de acceso y funcionalidad de usuarios al sitio, de identificar al usuario antes de que pueda realizar cualquier acción sobre el sistema. Garantiza que la información sea vista únicamente por quien tiene derecho a verla.

Existe un primer nivel o nivel básico donde están las funciones asociadas al usuario general o común, que requieren poca responsabilidad, en este nivel están los usuarios que solo verán datos del sistema.

El segundo nivel está compuesto por funciones de mayor complejidad y que pueden destruir información relacionada a las entidades del sistema.

Se usan mecanismos de encriptación (MD5) de los datos que por cuestiones de seguridad no deben viajar al servidor en texto claro, como es el caso de las contraseñas.

Se hacen validaciones de la información tanto en el cliente como en el servidor, no obstante los usuarios acceden de manera rápida y operativa al sistema sin que los requerimientos de seguridad se conviertan en un retardo para ellos.

• Restricciones en el diseño y la implementación:

Es una aplicación web desarrollada con la tecnología para creación de páginas web dinámicas PHP 5 y base de datos en PostgreSQL.

• Legales:

El sistema sirve de cliente de administración a una UDDI y es quien implementa el estándar que se rige por normas internacionales y cumple con las normas y leyes establecidas en nuestro país. Las tecnologías escogidas para el desarrollo de la aplicación, están basadas en la licencia GNU/GPL.

• Confiabilidad:

La herramienta de implementación a utilizar tiene soporte para recuperación ante fallos y errores.

- **Software:**

Para el funcionamiento del sistema en el servidor será necesario el SO Windows 98 o superior, Linux o Unix, en sus versiones de SO servidores. Para el funcionamiento del sistema en las terminales cliente será necesario el SO Windows 98 o superior, Linux o Unix.

- **Hardware:**

Se necesitan como requerimientos mínimos una PC con procesador Pentium II o superior.

- **Confidencialidad:**

Toda la información está protegida del acceso no autorizado, los administradores de sistema son los únicos que podrán gestionar cualquier información, por su parte los clientes solo podrán realizar búsquedas sobre la información de los servicios web.

- **Disponibilidad:**

Se garantiza a los usuarios del sistema el acceso a la información solicitada en todo momento (si tiene permiso para ello).

- **Restricciones:**

Se utiliza UML para lograr una mejor documentación del sistema y como herramienta de apoyo Visual Paradigm. Se utiliza como lenguaje de programación PHP 5 y el gestor de base de datos PostgreSQL. Como sistema de gestión de contenido se utiliza Drupal.

- **Ayuda y documentación en línea:**

Cuando el software se implemente, el producto brindará a los usuarios una buena ayuda en línea de modo que si el usuario presenta algún problema cuando vaya a realizar una búsqueda específica, pueda acudir al mismo, así como una documentación apropiada para el mejor trabajo con el mismo.

2.5 Definición de los Casos de Uso del Sistema.

2.5.1 Actores del Sistema.

Los actores del sistema pueden representar el rol que juega una o varias personas, un equipo o un sistema automatizado, como parte del sistema, y pueden intercambiar información con él o ser recipientes pasivos de información.

En este caso los actores que interactúan con el sistema se definen a continuación.

ACTORES DEL SISTEMA	DESCRIPCIÓN
Usuario	Individuo que sin registrarse busca cierto servicio web en el Nodo Privado UDDI, y utiliza los servicios que están contenidos en el sistema.
Publicador	Individuo que se registra y puede buscar, publicar, y configurar datos de los servicios del Nodo Privado UDDI. Además puede ver y modificar datos almacenados de los servicios del Nodo Privado UDDI.
Administrador	Individuo que se registra y puede buscar, publicar, coordinar, y administrar los servicios del Nodo Privado UDDI. Además puede administrar opciones de servicio y manejar ajustes de la seguridad.

Tabla 2.1 Identificación de autores.

2.5.2 Listado de los Casos de Uso del Sistema.

CU-1	Autenticar usuarios.
Actor	Usuario.
Descripción	El usuario accede al formulario de autenticación en el que llenará los campos de usuario y contraseña, el sistema comprobará la identidad y si es correcta podrá acceder al sitio de forma personalizada, de lo contrario, se le dará la posibilidad de volver al formulario. Es necesario autenticarse en el caso del publicador y el administrador, para que puedan cumplir con las actividades que realizan en la aplicación.
Referencia	RF1.

CU-2	Buscar información de servicio web.
Actor	Usuario.
Descripción	Sin tener que autenticarse el usuario puede hacer búsquedas de servicios, de proveedores y de especificaciones técnicas.
Referencia	RF2.

CU-3	Gestionar la información de servicios web.
Actor	Publicador.
Descripción	El usuario puede buscar, modificar, editar, registrar y eliminar información acerca de proveedores y especificaciones técnicas.
Referencia	RF3, RF1

CU-4	Administrar sistema.
Actor	Administrador.
Descripción	El usuario administra opciones de servicio, además de los datos en el sistema. Es el encargado de administrar y mantener el sistema.
Referencia	RF4, RF1.

CU-5	Administrar usuarios.
Actor	Administrador.
Descripción	El usuario administra los usuarios del sistema, es el encargado de registrar, eliminar, y modificar usuarios.
Referencia	RF5, RF1.

CU-6	Controlar acceso a la información.
Actor	Administrador.
Descripción	El usuario administra toda la información del sistema, controlando así el acceso a la misma, para esto define los niveles necesarios para que los usuarios puedan acceder a la información, además puede añadir, eliminar y modificar un rol.
Referencia	RF6, RF5, RF1.

2.5.3 Diagrama de Casos de Uso del Sistema.

DIAGRAMA DE CASOS DE USO DEL SISTEMA

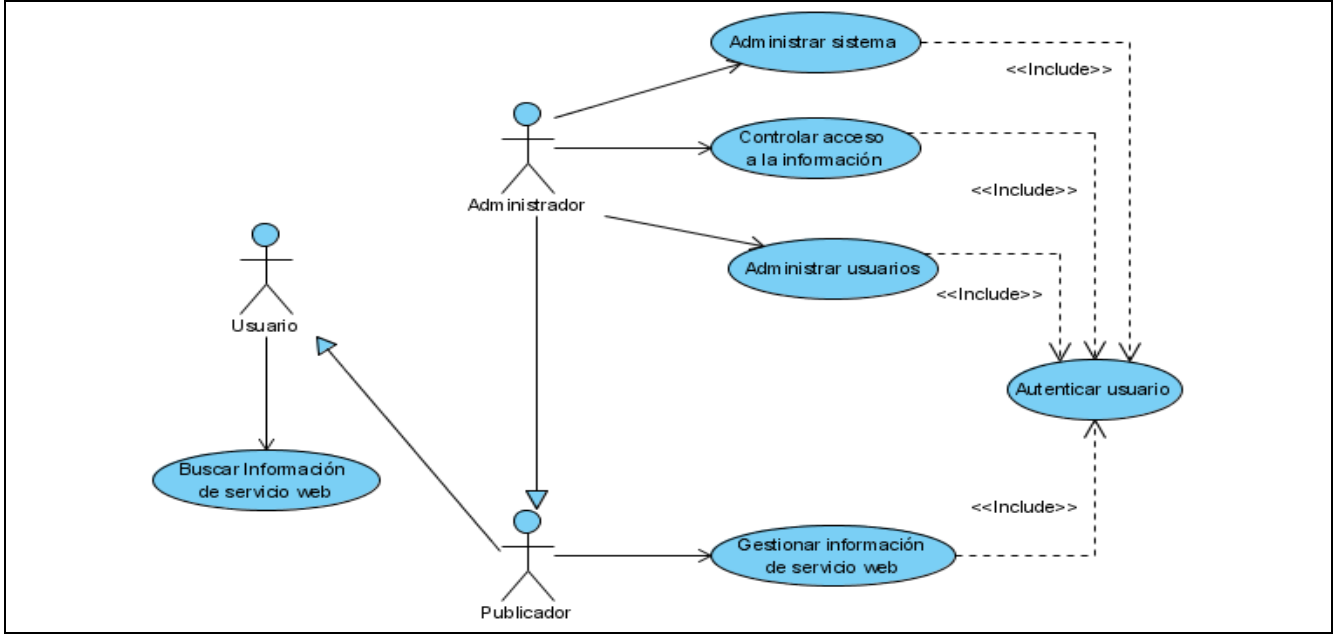


Figura 2.2 Diagrama Casos de Usos del Sistema.

2.5.4 Descripción de los Casos de Uso del Sistema.

Caso de Uso	**Autenticar Usuario.**
Actores	**Usuario** (inicia).
Propósito	Comprobar las credenciales de un usuario y autentificarlo en el sistema.
Resumen	El caso de uso inicia cuando el usuario accede a la página de autenticación, el sistema pregunta por las credenciales (usuario y contraseña), chequea que éstas sean válidas y autentifica al usuario para que pueda tener acceso al sistema.
Referencias	RF1
Precondiciones	-
Poscondiciones	El usuario queda autenticado (Queda registrado en una variable de sesión).
Curso Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1.-El usuario accede a la interfaz para autenticarse.	2. - El sistema muestra una interfaz con los siguientes datos: usuario y contraseña.
3.-El usuario introduce los datos.	4.-El sistema valida los datos. 5.-El sistema comprueba el acceso del usuario. 6.-El sistema muestra la página de acceso del usuario.
Curso Alterno de Eventos	
Acción del Actor	**Respuesta del Sistema**
3.-El usuario introduce los datos.	4. -El sistema valida los datos. 5.-El sistema comprueba el acceso del usuario. 6.-El sistema muestra la página de acceso del usuario según el rol del mismo con los permisos que le corresponde.

Prioridad:	Crítico.

Tabla 2.2 Descripción CU _ Autenticar Usuario.

Caso de Uso	**Buscar Información de servicio web.**
Actores	**Usuario.**
Propósito	Permitir buscar servicios, proveedores o especificación técnica.
Resumen	En este caso de uso el usuario sin necesidad de autenticarse podrá buscar información acerca de los servicios web, esta información la puede hacer mediante la búsqueda de servicios, mediante la búsqueda de proveedores o mediante la búsqueda de especificación técnica. En todos los casos el sistema devolverá un listado de acuerdo a lo que se ésta buscando.
Referencias	RF2.
Precondiciones	-
Poscondiciones	1.- Información del servicio buscada, mostrada al usuario. 2.- Información de especificación técnica buscada, mostrada al usuario. 3.- Información de proveedores buscada, mostrada al usuario.
Curso Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1.- El usuario selecciona una de las opciones de la interfaz, en este caso, Buscar.	2.- El sistema ejecuta alguna de las siguientes acciones: a) Si decide buscar servicios, ir a la sección: — “Buscar servicio”. b) Si decide buscar proveedores, ir a la sección: — “Buscar proveedores”.

	c) Si decide buscar especificación técnica, ir a la sección —"Buscar especificación técnica".
Curso Alterno de Eventos	
Sección "Buscar servicio".	
Acción del Actor	**Respuesta del Sistema**
	3.- El sistema muestra una interfaz con los siguientes parámetros de búsquedas: - Nombre del servicio. - Servicios por especificaciones técnicas.
4.- El usuario selecciona según lo que desea buscar. 5.- El usuario llena los datos.	6.- El sistema valida los datos. 7.- El sistema busca los servicios. 8.- El sistema muestra el resultado de la búsqueda que contiene los datos. - Nombre del servicio. - Lista de servicios por especificaciones técnicas.
Sección "Buscar proveedores".	
Acción del Actor	**Respuesta del Sistema**
	3.- El sistema muestra una interfaz con los siguientes parámetros de búsquedas: - Nombre del proveedor. - Detalle especificaciones técnicas.
4.- El usuario selecciona según lo que desea buscar. 5.- El usuario llena los datos.	6.- El sistema valida los datos. 7.- El sistema busca los proveedores. 8.- El sistema muestra el resultado de la búsqueda que contiene los datos. - Nombre del proveedor. - Lista de proveedores por especificaciones técnicas.

Sección "Buscar especificación técnica".	
Acción del Actor	**Respuesta del Sistema**
	3.- El sistema muestra una interfaz con los siguientes parámetros de búsquedas: - Nombre de la especificación técnica.
4.- El usuario llena los datos.	6.- El sistema valida los datos. 7.- El sistema busca las especificaciones técnicas. 8.- El sistema muestra el resultado de la búsqueda que contiene los datos. - Nombre de la especificación.
Prioridad:	Crítico.

Tabla 2.3 Descripción CU_Buscar Información de Servicio Web.

Caso de Uso	**Administrar Sistema.**
Actores	**Administrador.**
Propósito	Permitir la administración del sistema.
Resumen	El caso de uso inicia cuando el administrador intenta acceder al módulo de administración, en este momento el sistema chequea que el usuario tenga permisos para la operación que solicita, si los tiene, el sistema se muestra en modo de administración. En este modo el sistema puede ser editado a plenitud. Además el administrador vela por el mantenimiento del sistema.
Referencias	RF4, RF1.
Precondiciones	El sistema se puede administrar en todo momento.
Poscondiciones	Debe existir un usuario autenticado con permisos para administrar el sistema.
Curso Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1.- El usuario selecciona una de las	2. - El sistema muestra el módulo de administración.

opciones de la interfaz, en este caso, administrar sistema.	
3.- El administrador del sistema accede a la opción que desea modificar.	4.- El sistema muestra la información correspondiente a la opción seleccionada.
5.- El administrador del sistema realiza las operaciones deseadas.	6.- El sistema guarda los cambios realizados.
Prioridad:	Crítico.

Tabla 2.4 Descripción CU_Administrar Sistema.

Caso de Uso	**Administrar usuarios.**
Actores	**Administrador.**
Propósito	Permitir la administración y la gestión de los usuarios del sistema.
Resumen	El caso de uso inicia cuando el administrador intenta acceder al módulo de administración, en este momento el sistema chequea que el usuario tenga permisos para la operación que solicita, si los tiene, el sistema se muestra en modo administración. Una vez que el sistema se encuentre en modo de administración el administrador, es el encargado de registrar, modificar o eliminar usuarios del sistema.
Referencias	RF5, RF1.
Precondiciones	En el sistema se puede administrar usuarios en todo momento.
Poscondiciones	1.- Debe existir un usuario autentificado con permisos para administrar usuarios. 2.- Información del usuario creado, registrada en la base de datos. 3.- Información del usuario eliminado, actualizada en la base de datos. 4.- Información del usuario modificado, actualizada en la base de datos.

Curso Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
El administrador puede necesitar: -Añadir un usuario, ir a la sección" Registrar usuario". -Eliminar un usuario, ir a la sección" Eliminar usuario". -Modificar la cuenta de un usuario, ir a la sección" Modificar usuario".	
Curso Alterno de Eventos	
Sección "Registrar usuario".	
Acción del Actor	**Respuesta del Sistema**
1. El administrador va a la sección de administración global de la aplicación y escoge dentro del panel de gestión de usuarios, la opción: "Usuarios".	2.- Muestra una página con todos los usuarios de la aplicación y la opción: "Añadir usuario".
3. El administrador escoge la opción: "Añadir Usuario".	4. El sistema muestra un formulario con los datos que debe llenar para adicionar el usuario.
5.- El administrador llena los campos obligatorios y presiona el botón: "Crear cuenta nueva".	6.- Procesa y guarda la cuenta del nuevo usuario en la base de datos, mostrando un mensaje, donde se expone que la cuenta ha sido creada satisfactoriamente y finaliza el caso de uso.
Curso Alterno de Eventos	
Sección "Eliminar usuario".	
Acción del Actor	**Respuesta del Sistema**
1. El administrador va a la sección de administración global de la aplicación y escoge dentro del panel de gestión de usuarios, la opción: "Usuarios".	2.- Muestra todos los usuarios del sistema con la opción de editar las cuentas.
3. El administrador busca el usuario que desee eliminar y escoge la opción editar.	4. El sistema muestra una página con los datos del usuario y la opción de eliminarlo.

5.- El administrador presiona el botón: "Eliminar".	6.- El sistema muestra un mensaje para asegurarse de que el administrador desea ejecutar la operación solicitada.
7.- El administrador está seguro de que desea eliminar el usuario y presiona el botón: "Aceptar".	9.- El sistema elimina el usuario y guarda los cambios, de esta manera finaliza el caso de uso.
Curso Alterno de Eventos	
Sección "Modificar usuario".	
Acción del Actor	**Respuesta del Sistema**
1. El administrador va a la sección de administración global de la aplicación y escoge dentro del panel de gestión de usuarios, la opción: "Usuarios".	2.- El sistema muestra todos los usuarios del sistema con la opción de editar las cuentas.
3. El administrador busca el usuario al que desea modificarle la cuenta y escoge la opción editar.	4. El sistema muestra una página con los datos del usuario habilitados, de manera que puedan ser modificados.
5.- El administrador modifica los campos que desea y presiona el botón: "Enviar".	6.- Procesa y guarda la cuenta en la base de datos y finaliza el caso de uso.
Prioridad:	Crítico.

Tabla 2.5 Descripción CU_Administrar Usuarios.

Caso de Uso	**Controlar acceso a la información.**
Actores	**Administrador.**
Propósito	Permitir la administración del sistema y controlar el acceso a la información.
Resumen	El caso de uso inicia cuando el administrador intenta acceder al módulo de administración, en este momento el sistema chequea que el usuario tenga permisos para la operación que solicita, si los tiene, el sistema se muestra en modo administración, permitiéndole la posibilidad de editar contenido y controlar acceso a la información, además de modificar, añadir y eliminar rol.
Referencias	RF6, RF5, RF1.

Precondiciones	En el sistema se puede controlar la información y gestionar los roles en todo momento.
Poscondiciones	Debe existir un usuario autenticado con permisos para gestionar los roles de los usuarios.
Curso Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1.- El usuario selecciona una de las opciones de la interfaz, en este caso, controlar acceso a la información.	2. - El sistema ejecuta las siguientes acciones: a) Si decide Añadir un rol, ir a la sección" Agregar rol". b) Si decide Eliminar un rol, ir a la sección" Eliminar rol". c) Si decide Modificar un rol, ir a la sección" Modificar un rol". d) Si decide Editar permisos, ir a la sección" Editar permisos".
Curso Alterno de Eventos	
Sección "Agregar rol".	
Acción del Actor	**Respuesta del Sistema**
3.- El administrador va a la sección de administración global de la aplicación y escoge dentro del panel de gestión de usuarios, la opción: "Rol".	4.- Muestra una página con todos los roles y un cuadro de texto en el cual se debe poner el nombre del rol a agregar.
5.- Introduce el nombre del rol que desea agregar y presiona el botón: "Añadir rol".	6.- Guarda el rol en la base de datos y termina el caso de uso.
Sección "Modificar rol".	
Acción del Actor	**Respuesta del Sistema**
3.- El administrador va a la sección de administración global de la aplicación y escoge dentro del panel de gestión de usuarios, la opción: "Rol".	4.- Muestra todos los roles del sistema con las siguientes opciones: -Editar rol. -Editar Permisos (ir a la sección "Editar permisos").
5.- El administrador busca el rol que desea modificar y escoge la opción "Editar".	6.- Muestra una página con el nombre del rol habilitado para que pueda ser modificado, con las opciones: -Guardar rol.

	-Elimina rol (ir a la sección “Eliminar rol”).
7. El administrador modifica el nombre y presiona el botón: “Guardar rol”.	8.- Procesa y guarda el rol en la base de datos y finaliza el caso de uso.
Sección “Editar permisos”.	
Acción del Actor	**Respuesta del Sistema**
3.- El administrador escoge la opción: “Editar permisos”.	4.- Muestra una página con todos los permisos que pueden ser asignados a ese rol de acuerdo al contenido existente en la aplicación.
5.- Asigna los permisos que crea conveniente y presiona el botón: “Guardar permisos”.	6.- Guarda los cambios en la base de datos y termina el caso de uso.
Sección “Eliminar rol”.	
Acción del Actor	**Respuesta del Sistema**
3.- El administrador escoge la opción: “Eliminar rol”.	4.- Muestra un mensaje para asegurarse que el administrador desea eliminar el rol.
5.- Está seguro de que desea eliminar el rol y presiona el botón: “Aceptar”. 7.- Si el administrador no está seguro de si desea eliminar el rol escoge la opción: “Cancelar”.	6.- Elimina el rol y termina el caso de uso. 8.- No se ejecuta ninguna acción.
Prioridad:	Crítico.

Tabla 2.6 Descripción CU_Controlar acceso a la información.

Caso de Uso	**Gestionar la Información de servicio web.**
Actores	**Publicador.**
Propósito	El usuario puede buscar, modificar, editar, registrar y eliminar información acerca de proveedores y especificación técnica.
Resumen	En este caso de uso el usuario podrá gestionar información acerca de los servicios web, o sea, publicarla, esta

	información sería de proveedores o de especificaciones técnicas. En todos los casos, el sistema permitirá listar, adicionar y eliminar información.
Referencias	RF3, RF1.
Precondiciones	El usuario debe tener permisos para acceder a esta parte del sistema.
Poscondiciones	1.- Información de los proveedores, publicada en el sistema. 2.- Información de las especificaciones técnicas, publicadas en el sistema.
Curso Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1.- El usuario selecciona una de las opciones de la interfaz, en este caso, Publicar.	2.- El sistema ejecuta alguna de las siguientes acciones: a) Si desea gestionar información de proveedores, ir a la sección: — "Publicar proveedores". b) Si desea gestionar información de las especificaciones técnicas, ir a la sección: —"Publicar especificaciones técnicas".
Curso Alterno de Eventos	
Sección "Publicar proveedores".	
Acción del Actor	**Respuesta del Sistema**
	3.- El sistema muestra una interfaz con los siguientes parámetros de publicación: - Ver proveedor existente. - Eliminar proveedor existente. - Adicionar proveedor.
4.-El usuario selecciona ver proveedor existente o adicionar proveedor.	5.- El sistema valida los datos. 6- El sistema muestra las siguientes opciones: a) Detalles. b) Servicios c) Contacto d) Identificadores. e) Categorías.

	f) URL. g) Relaciones.
7.-El usuario selecciona la opción según lo que desea publicar.	8.-El sistema valida los datos. 9.- El sistema guarda los datos.
10.- El usuario selecciona eliminar proveedor existente.	11.- El sistema muestra un cartel para confirmar la acción de eliminar.
12.- El usuario da clic en aceptar.	13.- El sistema elimina el proveedor. 14.- El sistema guarda los cambios.
Sección "Publicar especificación técnica".	
Acción del Actor	**Respuesta del Sistema**
	3.- El sistema muestra una interfaz con los siguientes parámetros de publicación: - Ver especificación técnica existente. - Adicionar especificación técnica. - Eliminar especificación técnica existente.
4.-El usuario selecciona ver especificación técnica existente o adicionar especificación técnica.	5.- El sistema valida los datos. 6- El sistema muestra las siguientes opciones: a) Detalles. b) Identificadores. c) Categorías. d) Documentos generales.
7.-El usuario selecciona la opción según lo que desea publicar.	8.-El sistema valida los datos. 9.- El sistema guarda los datos.
10.- El usuario selecciona eliminar especificación técnica existente.	11.- El sistema muestra un cartel para confirmar la acción de eliminar.
12.- El usuario da clic en aceptar.	13.- El sistema elimina la especificación. 14.- El sistema guarda los cambios.
Prioridad:	Crítico.

Tabla 2.7 Descripción CU_Gestionar la información de servicios web.

2.6 Conclusiones.

En este capítulo se comenzó a desarrollar la propuesta de solución, obteniéndose a partir del análisis del modelo del dominio, un listado con las funciones que debe tener el sistema, que se representaron

mediante un Diagrama de Casos de Uso. El sistema tiene 6 casos de usos, de ellos, 4 significativos y 2 complejos, el principal es el caso de uso Buscar la Información de servicios web, finalmente se describieron paso a paso todas las acciones de los 3 actores del sistema que se seleccionaron con los casos de uso con los que interactúan. Gracias a esto ahora se puede empezar a construir el sistema, tratando de que se cumplan todos los requerimientos y las funciones que han sido consideradas necesarias en este capítulo.

Capítulo

3

ANÁLISIS Y DISEÑO

3.1 Introducción.

Tras la definición y descripción, en el anterior capítulo, de las funcionalidades deseadas y necesarias del sistema propuesto; se hace necesario definir cómo se desarrollará. Este capítulo tiene el objetivo de plantear la concepción general del diseño del sistema propuesto. Así, se presentan los diagramas de clases web que detallan la interacción de las distintas páginas y se estructura la información que se desea persista a través del diseño de la base de datos. Son también descritos los estándares de diseño seguidos.

3.2 Modelo de Análisis.

El objetivo de realizar el modelo de análisis es conseguir una comprensión más precisa de los requisitos y una descripción de los mismos que sea fácil de mantener y que sirva de ayuda para estructurar el sistema entero.

En el análisis se razonan los requisitos con mayor profundidad y se utiliza el lenguaje del desarrollador para describir los resultados. Esto da la ventaja de inferir más sobre los aspectos internos del sistema, y por tanto resolver elementos relativos a la interferencia de casos de uso.

Esboza como llevar a cabo la funcionalidad dentro del sistema, incluida la funcionalidad significativa para la arquitectura, sirve como una primera aproximación al diseño.

El modelo de análisis está compuesto por clase de análisis, que representan una abstracción de una o varias clases y/o subsistemas del diseño del sistema. Posee un conjunto de características tales como: se centra en el tratamiento de los requisitos funcionales y pospone los no funcionales, define atributos y participa en relaciones, entre otras.

Cada clase posee uno de los estereotipos siguientes:

- **Clases de interfaz:** se utilizan para modelar la interacción entre el sistema y sus actores, la cual implica recibir (y presentar) información y peticiones desde (y hacia) los usuarios.
- **Clases de entidad:** se utilizan para modelar información que posee una vida larga o que es a menudo persistente, modelan la información y el comportamiento asociado de algún fenómeno o concepto, como una persona, un objeto, o un suceso del mundo real.
- **Clases de control:** Coordinan la realización de uno o más CU, coordinando las actividades de los objetos que implementan las funcionalidades del mismo.

Cumpliendo con lo anterior analizado y por la importancia en su información se definirán las entidades del modelo de análisis.

- **E_Proveedor:** entidad que contiene los datos del proveedor como el nombre del proveedor, descripción, URL del proveedor.
- **E_Contacto:** entidad que contiene los datos de los contactos de un proveedor como nombre del contacto, dirección.
- **E_Servicio:** entidad que contiene los datos de los servicios que ofrece un proveedor como son nombre del servicio, descripción.
- **E_Categoría:** entidad que contiene los datos de las categorías y sub categorías como por ejemplo el nombre.
- **E_Identificador:** entidad que contiene los datos que identifican a un proveedor y a una especificación técnica como nombre de identificador o valor.
- **E_Especificación Técnica:** entidad que contiene los datos de las especificaciones técnicas de un servicio como el nombre de la especificación, descripción, URL de la especificación y detalles.
- **E_Rol:** entidad que contiene los datos de los diferentes roles que se le asignan a cada usuario.

3.2.1 Caso de Uso: Autenticar usuario.

DIAGRAMA DE CLASES AUTENTICAR

Figura 3.1 Diagrama de clases de análisis del CU Autenticar.

3.2.2 Caso de Uso: Buscar información de servicio web.

DIAGRAMA DE CLASES BUSCAR

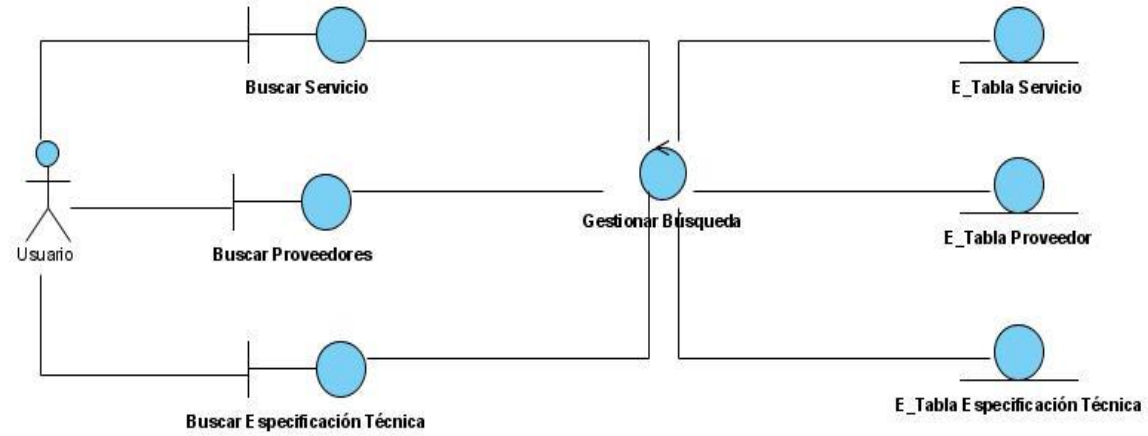


Figura 3.2 Diagrama de clases de análisis del CU Buscar información de servicio web.

3.2.3 Caso de Uso: Administrar sistema.

DIAGRAMA DE CLASES ADMINISTRAR SISTEMA

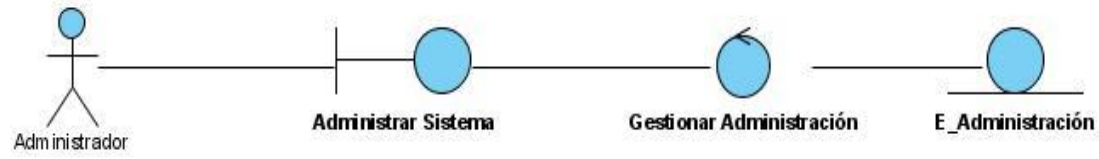


Figura 3.3 Diagrama de clases de análisis del CU Administrar Sistema.

3.2.4 Caso de Uso: Administrar usuarios.

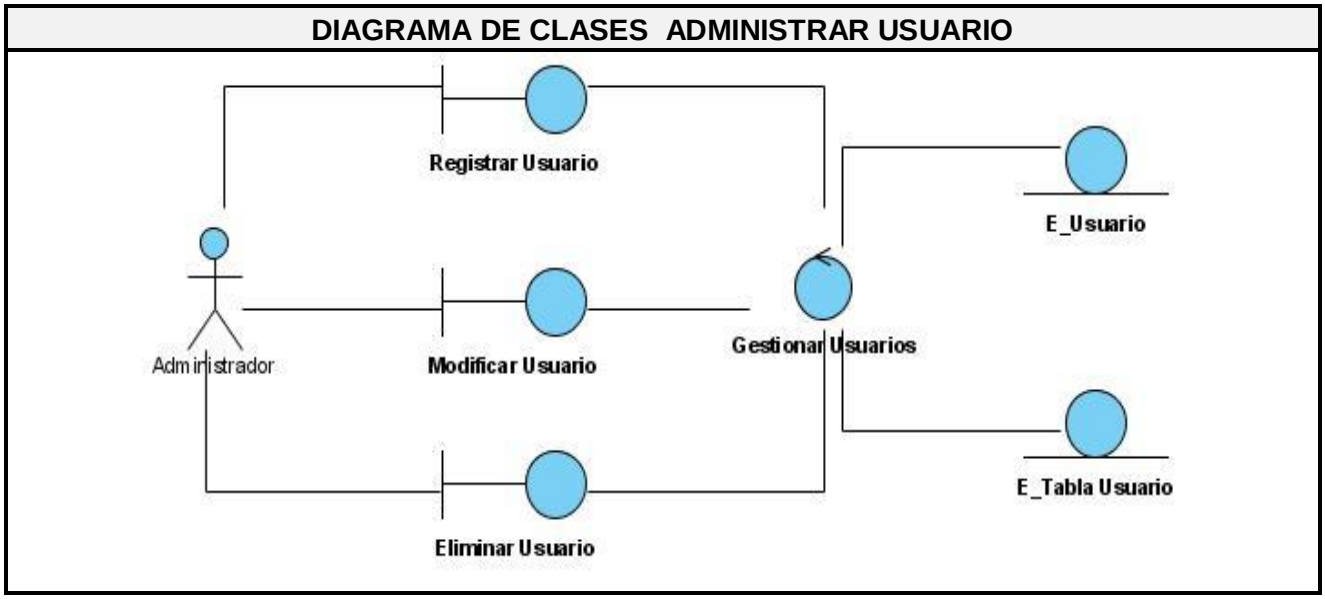


Figura 3.4 Diagrama de clases de análisis del CU Administrar Usuarios.

3.2.5 Caso de Uso: Controlar acceso a la información.

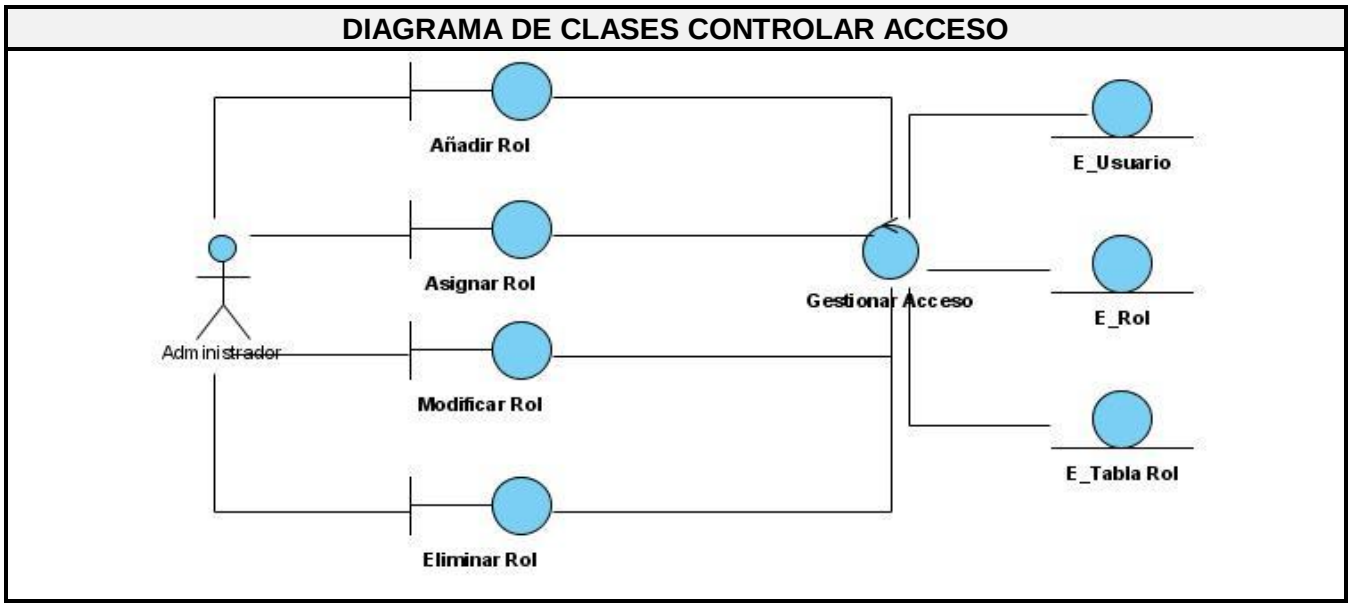


Figura 3.5 Diagrama de clases de análisis del CU Controlar acceso a la información.

3.2.6 Caso de Uso: Gestionar la información de servicios web.

DIAGRAMA DE CLASES GESTIONAR

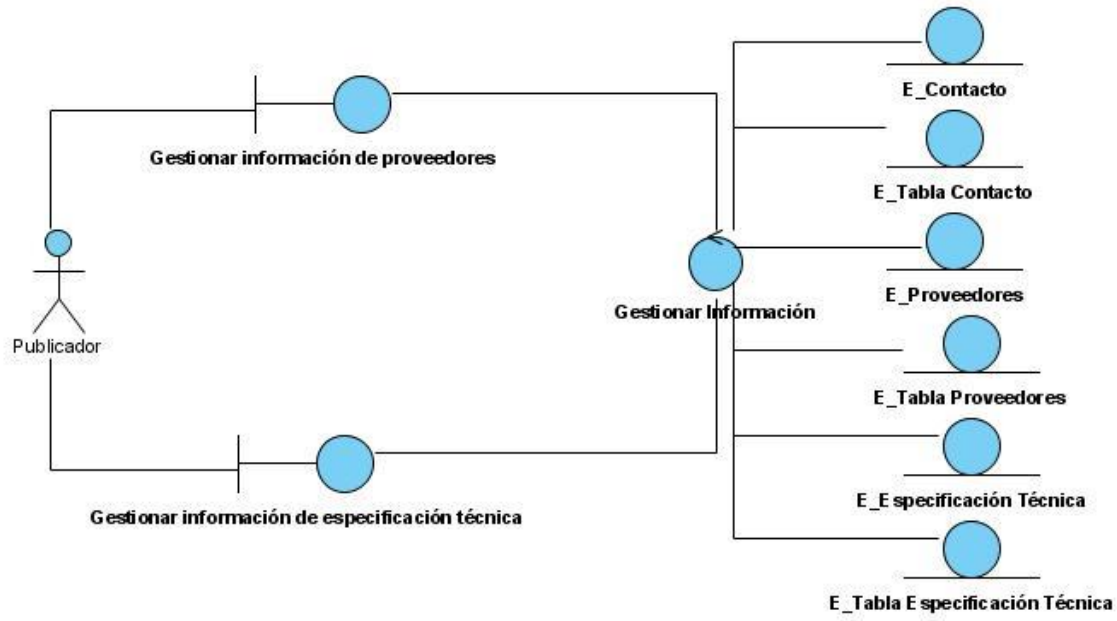


Figura 3.6 Diagrama de clases de análisis del CU Gestionar la información de servicios web.

3.3 Modelo de Diseño

El modelo del diseño es un modelo de objetos que describe la realización física de los casos de uso, centrándose en cómo los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema a considerar. Además el modelo de diseño sirve de abstracción de la implementación del sistema, y es de ese modo, utilizada como una entrada fundamental de las actividades de implementación y prueba.

3.3.1 Diagramas de Clases de Diseño Web.

Debido a la utilización de Drupal como CMS para desarrollar el sistema se explicarán brevemente el funcionamiento del mismo y posteriormente se desarrollará el diagrama de clases. Drupal genera el contenido indexado en tiempo real en forma de páginas llamadas Node. Estas páginas contienen las propiedades básicas de cualquier publicación, como por ejemplo: título, autor, fecha de creación y otras; las cuales pueden aumentar en la medida en que se integren nuevos módulos a la aplicación.

Un módulo, no es más que un archivo que cuenta con uno o más ficheros, el fichero principal es registrado bajo la extensión: .module e implementa una interface definida por el CMS. Existen dos categorías para los módulos:

- **Módulos de contenido:** Son los que definen un nuevo tipo de contenido, así como las funcionalidades para su creación, edición y posterior publicación.
- **Módulos funcionales:** Poseen múltiples propósitos en dependencia de las funcionalidades que se deseen implementar en un sistema determinado.

A continuación se muestran las secciones en las que se divide Drupal.

- **Paquete 1.1 Themes:** Contiene las plantillas del sistema.
- **Paquete 1.2 Includes:** Contiene los ficheros indispensables para el funcionamiento como por ejemplo: *Database,* que es el que provee las funcionalidades de acceso a la base de datos.
- **Paquete 1.3 Modules:** Incluye los módulos que facilitan las funcionalidades de la aplicación.
- **Paquete 1.4 Script:** Contiene los ficheros correspondientes al manejo visual del sistema, CSS y java script.

Aunque Drupal no es orientado a objetos, los scripts están organizados en paquetes, así se muestra a continuación.

3.3.1.1 Diagrama de clases del diseño web.

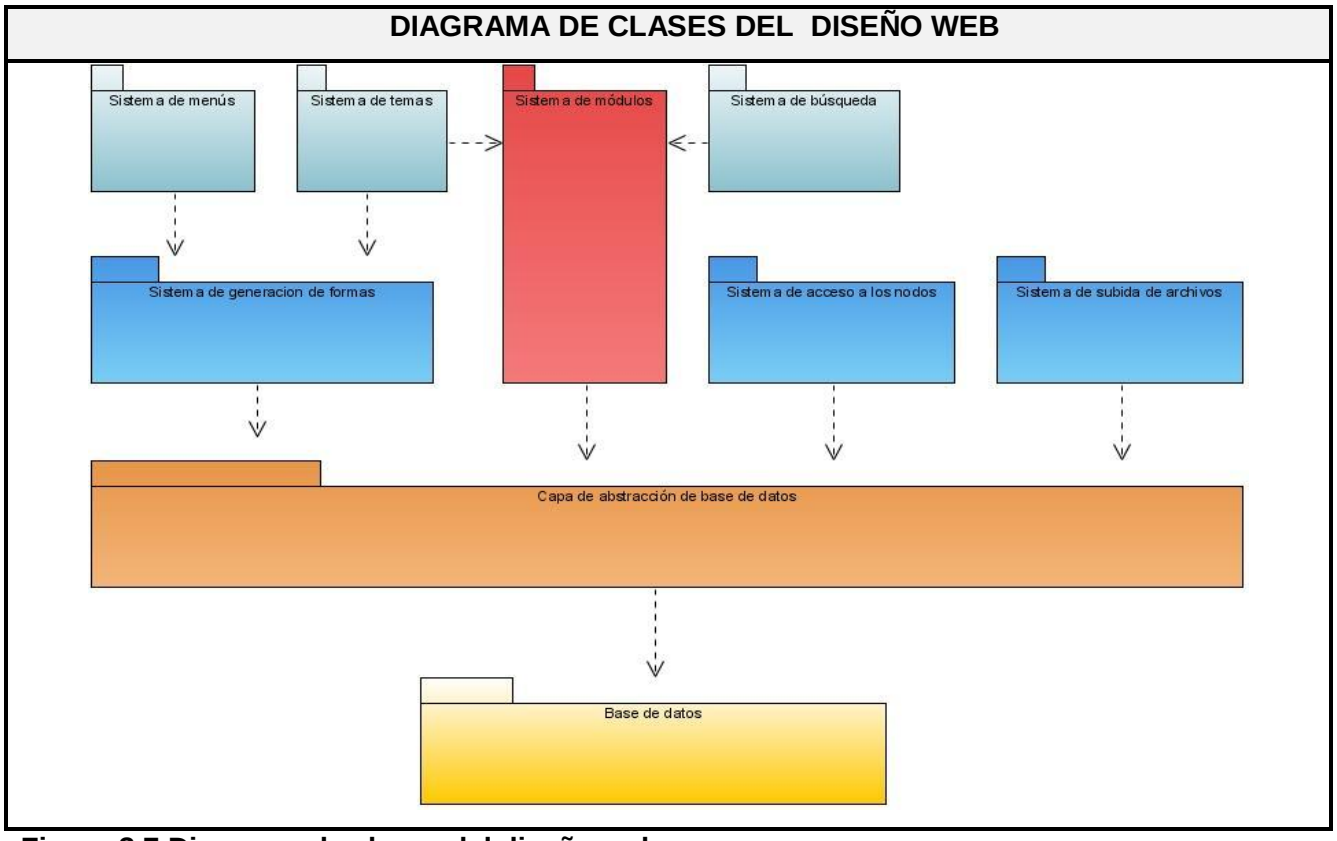


Figura 3.7 Diagrama de clases del diseño web.

3.3.1.2 Descripción de los módulos usados para el desarrollo del sistema.

Los módulos en Drupal son ficheros con extensión .module que contienen funciones escritas en PHP. Éstas son llamadas por Drupal durante sus procesos habituales de gestión de contenido. Por ejemplo, cada vez que un nodo es creado, visualizado, modificado o borrado, Drupal llama a una de estos *hooks* pasándoles el contenido del nodo. De esta forma los módulos tienen la posibilidad de modificar y adaptar la información a visualizar en las páginas web antes de que se mande definitivamente al navegador.

A continuación mostraremos los módulos usados para el desarrollo del sistema y su descripción.

MÓDULOS	DESCRIPCIÓN
Publish Providers	Permite insertar proveedores así como los datos correspondientes a estos como son Detalles, Servicios, Binding, Contactos, URLs, Identificadores. Además permite hacer búsquedas de proveedores y servicios por diferentes criterios.
Publish tModels	Permite insertar modelos técnicos así como los datos correspondientes a estos como son Detalles, Identificadores y Documentos Generales. Además permite hacer búsquedas de dichos modelos utilizando diferentes criterios.
Color	Permite que el usuario cambie el esquema de colores de algunos temas.
Help	Gestiona la visualización de la ayuda en línea.
Locale	Permite la traducción del sitio a varios idiomas.
Menú	Permite la creación de menús, este módulo es requerido por el módulo Nice Menus.
Path	Permite a los usuarios renombrar URLs.
Taxonomy	Activa la categorización del contenido.

3.3.2 Diagramas de interacción.

Los diagramas de interacción describen la forma en que cada operación detectada en los diagramas de ecuencia lleva a cabo sus responsabilidades y modifica el estado del sistema, mostrando el modo en que los objetos interactúan a través de mensajes. En UML los diagramas de interacción pueden representarse a través de los Diagramas de Colaboración y/o de los Diagramas de Secuencia, ambos son representaciones alternas de interacciones.

Los Diagramas de Secuencia destacan la ordenación temporal de los mensajes. Muestran la interacción entre objetos, ordenadas en secuencia temporal durante un escenario concreto. Los Diagramas de Colaboración muestran como los objetos se asocian unos con otros.

3.3.2.1 Diagramas de Secuencia.

3.3.2.1.1 Diagrama de secuencia Autenticar Usuario.

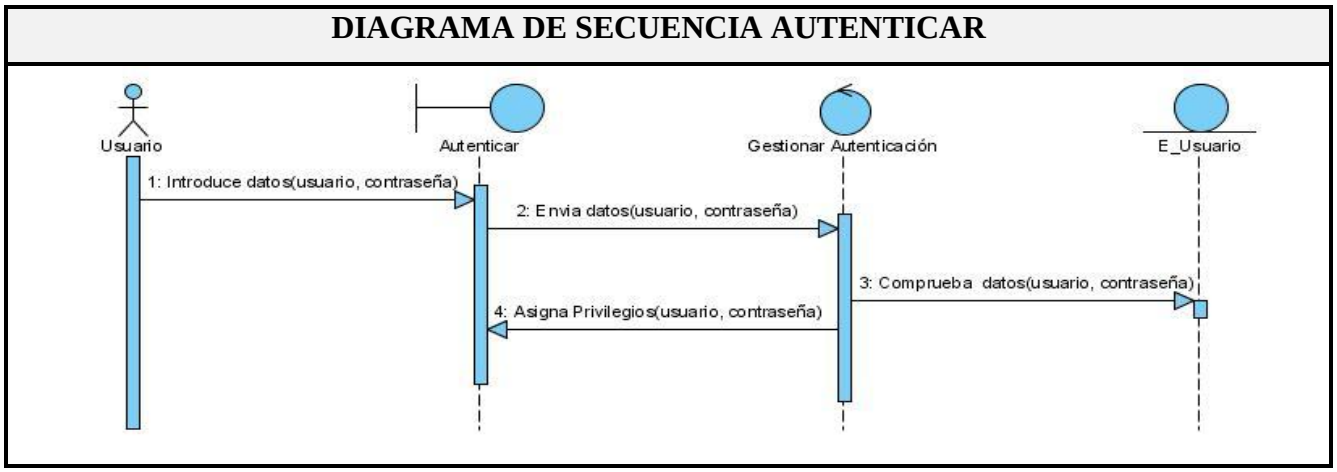


Figura 3.8 Diagrama de secuencia del CU Autenticar Usuario.

3.3.2.1.2 Diagrama de secuencia Buscar Servicio.

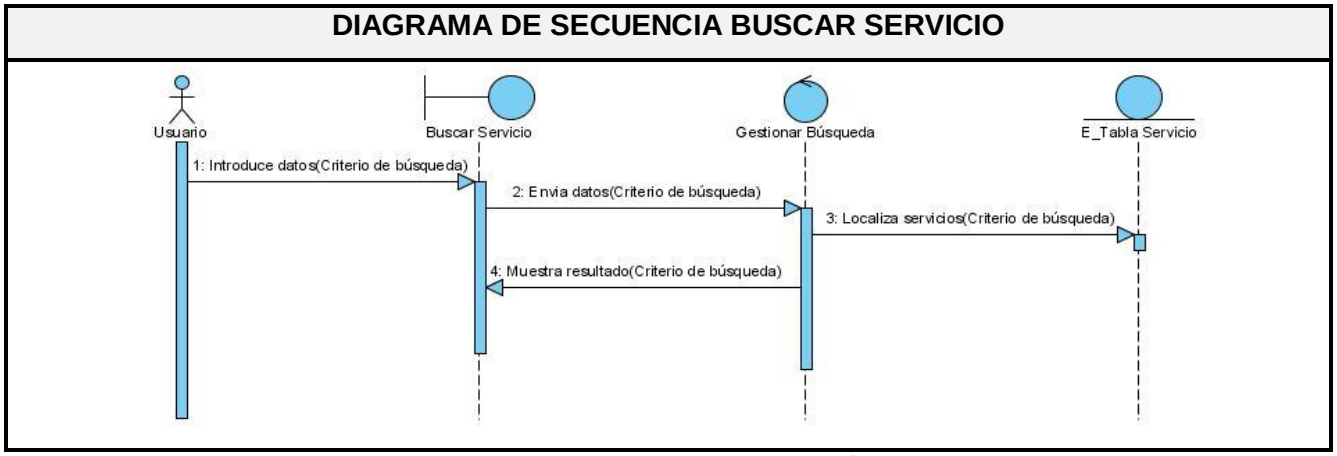


Figura 3.9 Diagrama de secuencia del CU Buscar información de servicio web.

3.3.2.1.3 Diagrama de secuencia Buscar Proveedores.

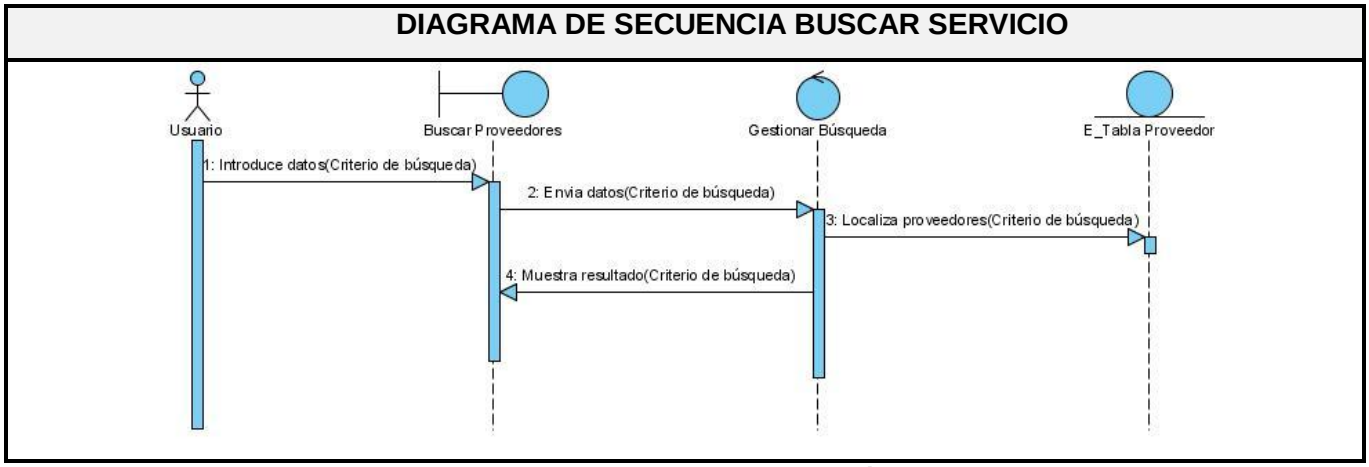


Figura 3.10 Diagrama de secuencia del CU Buscar información de servicio web.

3.3.2.1.4 Diagrama de secuencia Buscar Especificación Técnica.

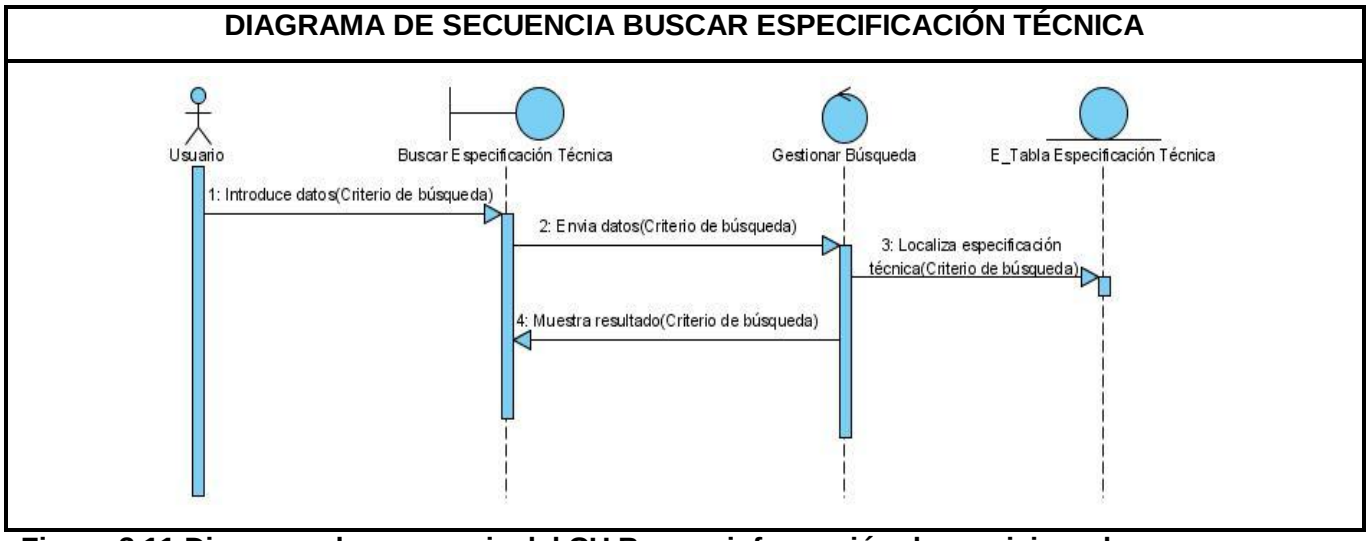


Figura 3.11 Diagrama de secuencia del CU Buscar información de servicio web.

3.3.2.1.5 Diagrama de secuencia Administrar Sistema.

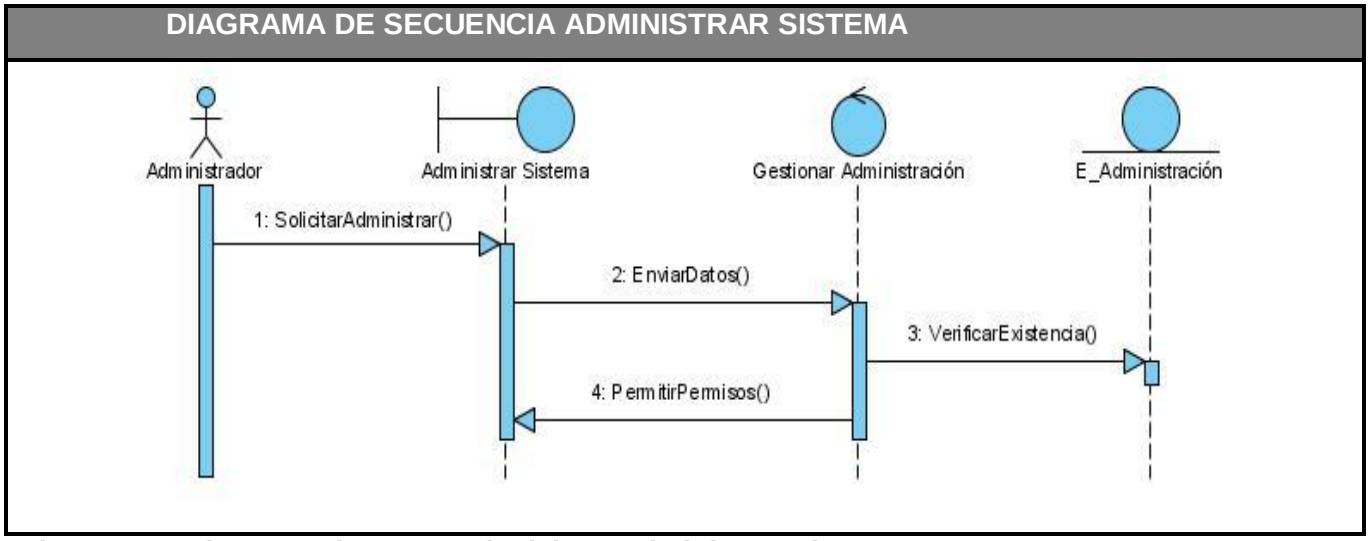


Figura 3.12 Diagrama de secuencia del CU Administrar Sistema.

3.3.2.1.6 Diagrama de secuencia Modificar Usuario.

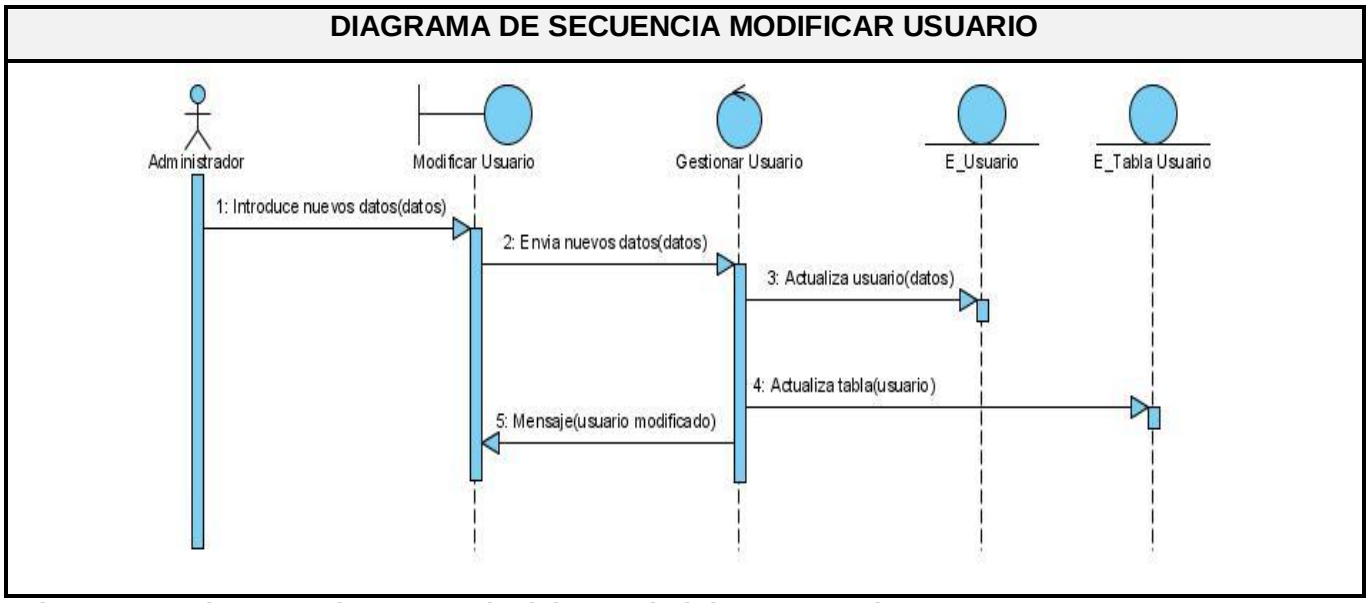


Figura 3.13 Diagrama de secuencia del CU Administrar Usuario.

3.3.2.1.7 Diagrama de secuencia Registrar Usuario.

DIAGRAMA DE SECUENCIA REGISTRAR USUARIO

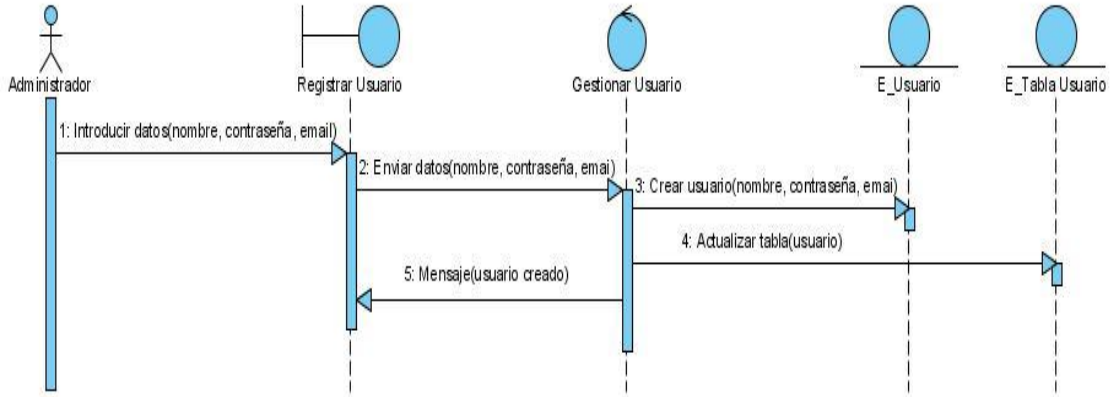


Figura 3.14 Diagrama de secuencia del CU Administrar Usuario.

3.3.2.1.8 Diagrama de secuencia Eliminar Usuario.

DIAGRAMA DE SECUENCIA ELIMINAR USUARIO

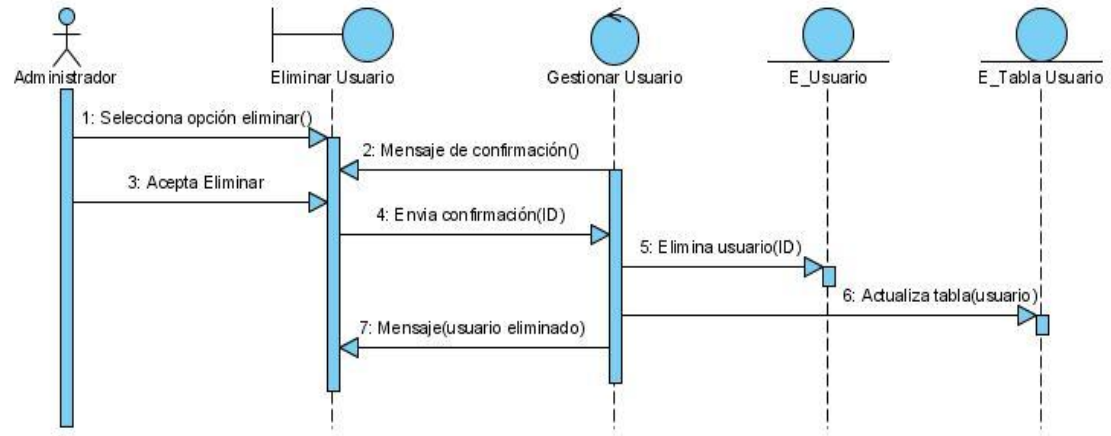


Figura 3.15 Diagrama de secuencia del CU Administrar Usuario.

3.3.2.1.9 Diagrama de secuencia Añadir Rol.

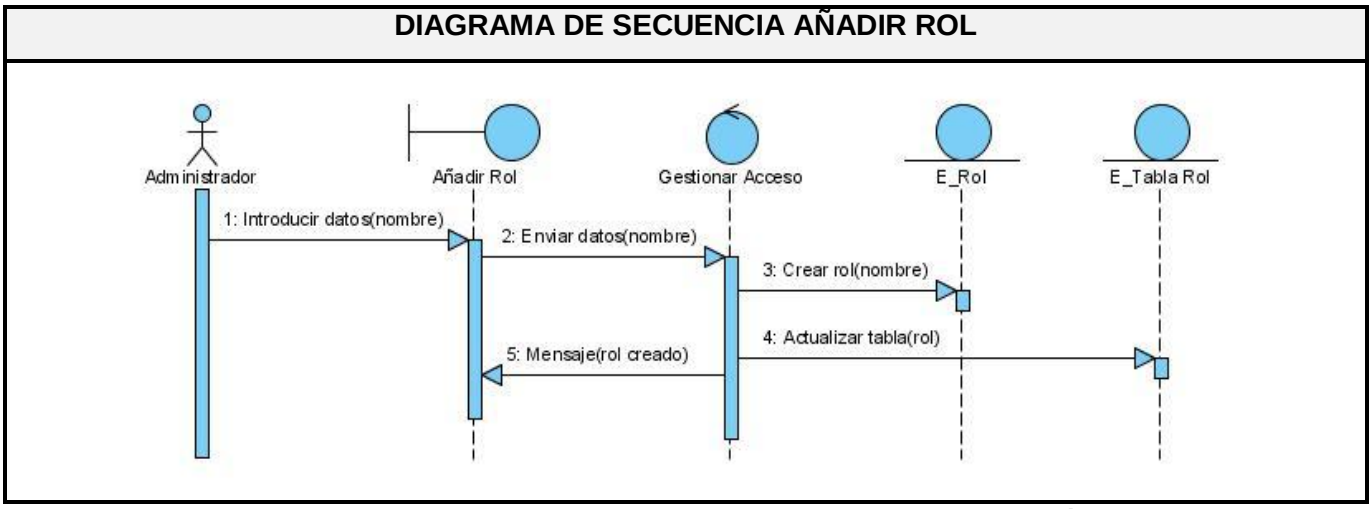


Figura 3.16 Diagrama de secuencia del CU Controlar Acceso a la Información.

3.3.2.1.10 Diagrama de secuencia Eliminar Rol.

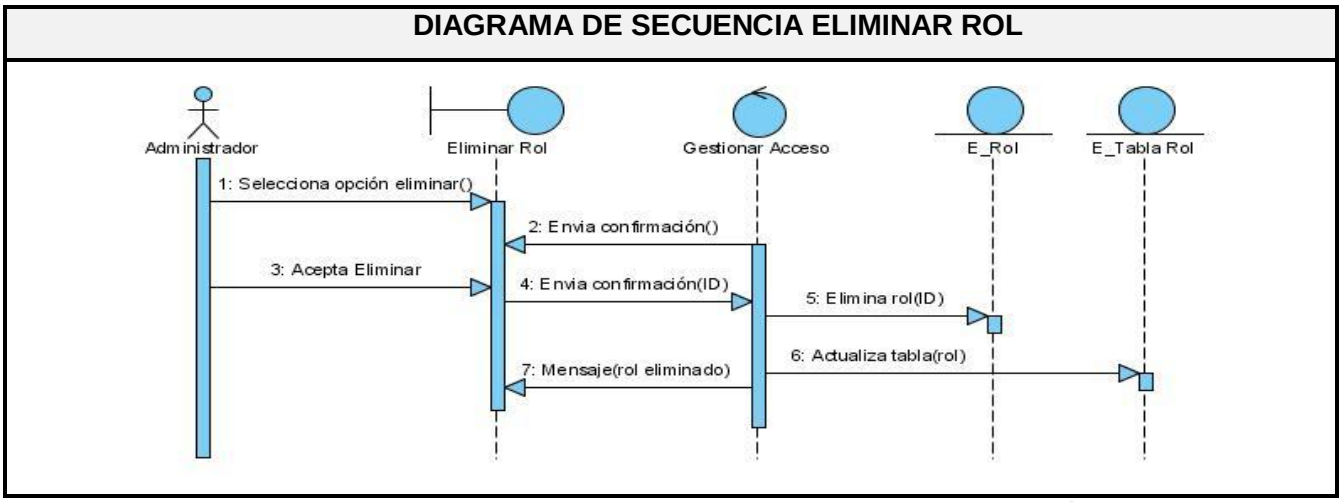


Figura 3.17 Diagrama de secuencia del CU Controlar Acceso a la Información.

3.3.2.1.11 Diagrama de secuencia Modificar Rol.

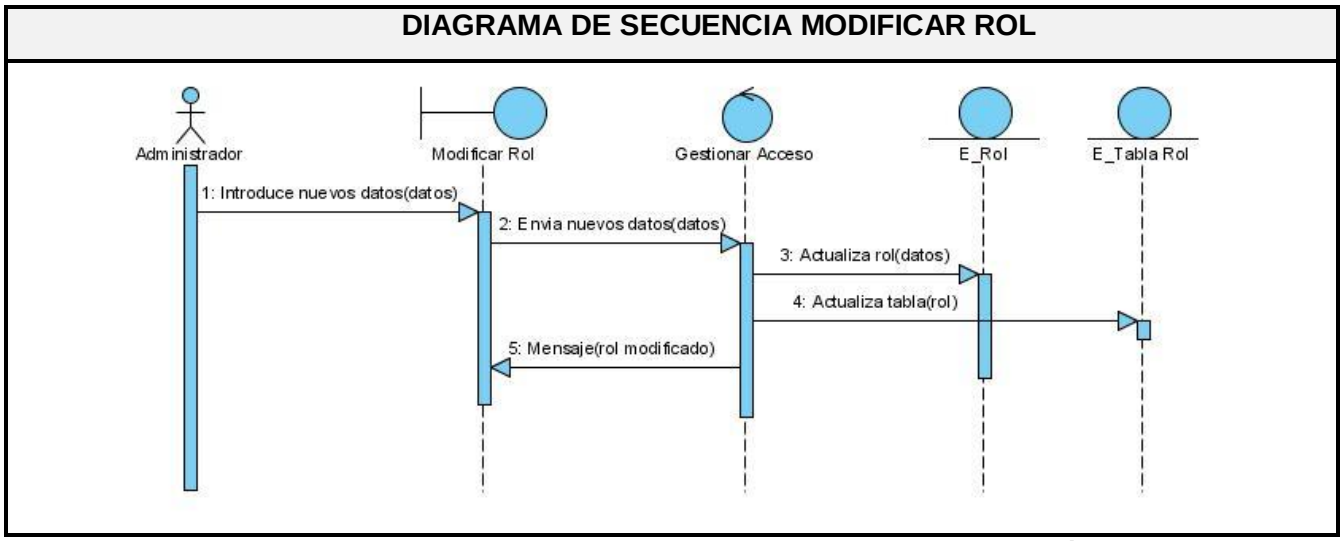


Figura 3.18 Diagrama de secuencia del CU Controlar Acceso a la Información.

3.4 Principios de Diseño.

El diseño de la interfaz de una aplicación, el formato de los reportes, la concepción de la ayuda y el tratamiento de excepciones tiene gran influencia en el éxito o fracaso de una aplicación. A continuación se describen los principios de diseño seguidos para el desarrollo del sistema.

3.4.1 Interfaz de usuario.

El diseño de interfaces de usuario es una tarea que ha adquirido relevancia en el desarrollo de un sistema, se puede definir como: —el conjunto de trabajos y pasos que seguirá el usuario, durante todo el tiempo que se relacione con el programa, detallando lo que verá y escuchará en cada momento, y las acciones que realizará, así como las respuestas que el sistema dará.

La calidad de la interfaz de usuario puede ser uno de los motivos que conduzca a un sistema al éxito o al fracaso, es por eso que uno de los aspectos más relevantes de la usabilidad de un sistema es la consistencia de su interfaz de usuario.

La aplicación UDDI fue diseñada para lograr una comunicación efectiva entre los usuarios, clientes y demás personas que necesiten satisfacer cualquier información acerca de los servicios que existen en la UCI. El diseño es sencillo, de fácil navegabilidad con la construcción de una plantilla en el Sistema de Gestión de Contenidos Drupal.

Todas las páginas web tienen una estructura similar, contienen una imagen identificativa del sistema.

Figura 3.21. Imagen de la parte superior de todas las páginas.

La aplicación debe estar disponible las 24 horas del día, en aras de lograr una alta disponibilidad, es conveniente comprender la forma en que las opciones de diseño ayudan a maximizar la disponibilidad de la aplicación, para ello se utilizó un número limitado de imágenes y animaciones para evitar largos tiempos de espera para poder visualizarlas. Además se permite el acceso a todas las secciones del sistema desde cualquier página, permitiendo una buena navegabilidad.

EL sistema posee un módulo de administración, el cual no presenta complejidad para el administrador del sistema. Basado en las funcionalidades de Drupal se proporciona a los publicadores y administradores de la aplicación editar contenido mediante una plataforma de edición y publicación potente y fácil de usar.

3.4.2 Tratamiento de errores.

El tratamiento de errores se realiza con el sistema de captura de errores de Drupal, una vez que ocurre una excepción el usuario es re direccionado a una página de error.

Cada formulario se encarga de la validación de sus datos para evitar errores de concepto. Y se utilizan mensajes de confirmación, para acciones que son irreversibles como por ejemplo:

Presencia de errores en forma de mensaje de texto en la misma página donde se ejecutó la acción, de forma que el usuario pueda corregir el error fácilmente.

Figura 3.22. Mensaje de error indicando que el campo contraseña no pude estar vacio.

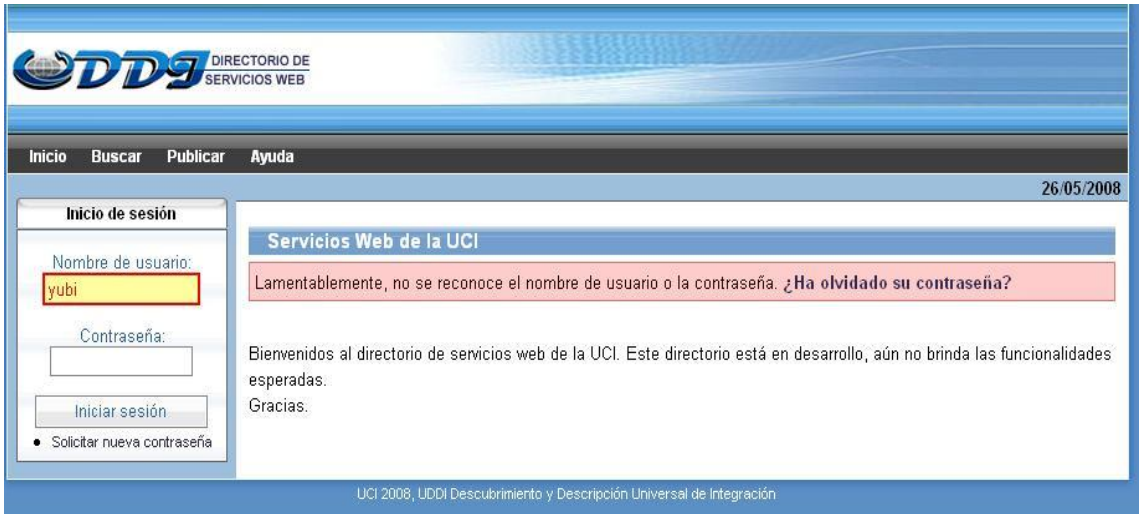


Figura 3.23. Mensaje de error indicando datos incorrectos.

Figura 3.24. Mensaje de error indicando campos vacios

3.4.3 Seguridad

Debido a las características de la Institución se hace necesario mantener de forma segura los datos para el óptimo funcionamiento del sistema.

En el sistema se puede realizar la búsqueda de información acerca de servicios, proveedores y modelos técnicos, por cualquier usuario de la institución, pero existen restricciones de acceso debido a la confidencialidad de algunos datos que solo pueden ser consultados por aquellos usuarios que se registren.

Cuando el usuario se autentica el sistema comprueba que las credenciales coincidan con las almacenadas y en caso de serlo muestra un menú de trabajo personalizado con los privilegios correspondientes según el tipo de usuario. En caso contrario se re direcciona nuevamente al usuario a la página principal y se le brinda la posibilidad de autenticarse de nuevo.

El administrador del sistema es el encargado de mantener el sistema, gestionar los usuarios y darle los niveles de acceso correspondientes a cada uno.

3.4.4 Concepción de la Ayuda.

Cada página de la aplicación muestra un texto, explicando cómo realizar las operaciones que esté realizando en ese momento el usuario, así como los campos son bastantes explícitos para mejorar la comprensión de lo que se debe introducir.

Además existe un sistema de Ayuda en la que se explica paso a paso como trabajar con el sistema.

3.5 Conclusiones.

En el presente capítulo se mostraron los resultados de la etapa de diseño del sistema. Se desarrollaron los diagramas de interacción a través de los diagramas de secuencia del diseño, para obtener una mayor comprensión del funcionamiento del sistema.

Con los resultados obtenidos en los diagramas de interacción y a modo de evolución del modelo conceptual, se obtuvo el diagrama de clases donde se representaron las clases de forma que quedó una representación de las clases, sus asociaciones, atributos y responsabilidades. Se describieron, además, los principios de diseño seguidos, específicamente, los temas de estándares de la interfaz, concepción del tratamiento de errores, sistema de ayuda y principios de codificación.

Todos estos elementos obtenidos brindan una idea mucho más clara de las clases, subsistemas y algoritmos para lograr una mejor codificación.

Capítulo

4

IMPLEMENTACIÓN Y PRUEBA

4.1 Introducción.

En el presente capítulo se desarrollan los flujos de trabajo de implementación y prueba. Se muestra la situación física de los componentes lógicos desarrollados a través del modelo de despliegue. Mediante el modelo de componentes se presenta la vista estática del sistema, además se muestra el modelo de prueba, donde se describen los casos de prueba de integración por cada caso de uso.

4.2 Diagrama de Despliegue.

El diagrama de despliegue es un modelo de objetos que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo. Es una colección de nodos y arcos; donde cada nodo representa un recurso de cómputo, normalmente un procesador o un dispositivo de hardware similar.

Muestra la configuración de los componentes hardware, los procesos, los elementos de procesamiento en tiempo de ejecución y los objetos que existen en tiempo de ejecución. En este tipo de diagramas intervienen nodos, asociaciones de comunicación, componentes dentro de los nodos y objetos que se encuentran a su vez dentro de los componentes. Un nodo es un objeto físico en tiempo de ejecución, es decir una máquina que se compone habitualmente de, por lo menos, memoria y capacidad de procesamiento, a su vez puede estar formada por otros componentes. El diagrama de despliegue muestra la topología del hardware sobre el que se ejecuta el sistema.

En el diagrama de despliegue se sitúa el software en el hardware que lo contiene. En este caso la aplicación se encuentra hospedada en un servidor web, la misma se comunica con un sistema de

gestión de base de datos (PostgreSQL) que se encuentra en otro servidor, también se comunica con otro servidor web (UDDI) mediante conexión HTTP, y por último mediante conexión SMTP se comunica con un servidor de correo.

Se emplea como protocolo de comunicación entre el cliente y el servidor web el HTTP debido a que está basado en el modelo cliente-servidor.

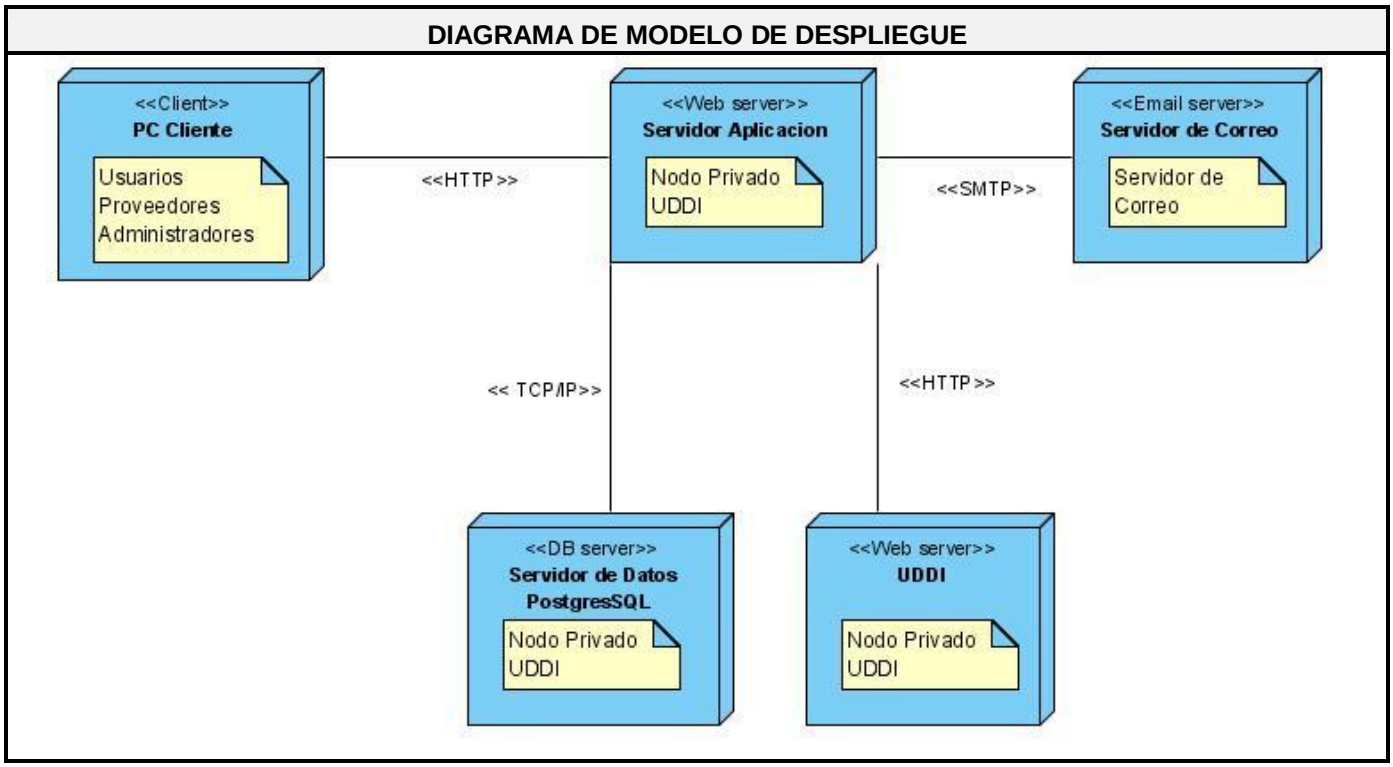


Figura 4.1 Diagrama de modelo de despliegue

4.3 Diagrama de componentes.

Los diagramas de componentes son usados para estructurar el modelo de implementación en términos de subsistemas de implementación y mostrar las relaciones entre los elementos de implementación.

Es un diagrama que muestra un conjunto de elementos del modelo tales como componentes, subsistemas de implementación y sus relaciones.

Es un grafo de componentes unidos a través de relaciones que pueden ser de compilación o de ejecución, y además se pueden representar las interfaces de esos componentes. Interfaces se refiere a la descripción de las operaciones que se realizan en esos componentes y en esos subsistemas.

Es otra forma de representar una vista estática del sistema, que representa la organización y dependencia que existiría entre los componentes físicos que se necesitan para ejecutar la aplicación, sean éstos componentes de código fuente, librerías, binarios o ejecutables. No es necesario que un diagrama incluya todos los componentes del sistema, normalmente se realizan por partes. Cada diagrama describe una parte del sistema.

DIAGRAMA DE COMPONENTES

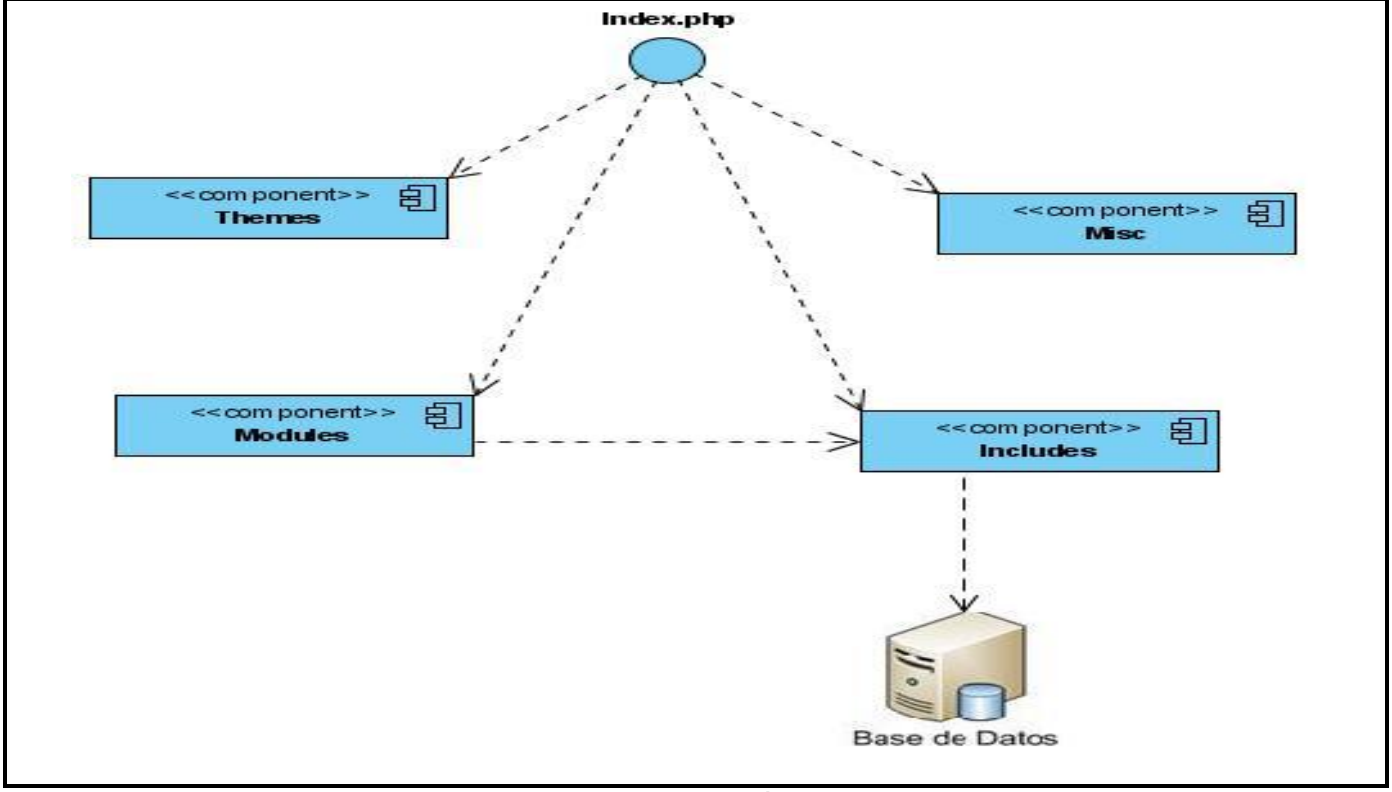


Figura 4.2 Diagrama de componentes A continuación una tabla que describe cada uno de los componentes representado en el diagrama anterior:

COMPONENTE	PROPOSITO
Index.php	Es el punto de inicio de la aplicación, a partir de esta entrada se solicitan los diferentes módulos del CMS.
Themes	Este componente incluye las plantillas que se pueden utilizar, nos permite separar el contenido de la presentación.
Modules	Este componente contiene todos los módulos, que permiten las distintas funcionalidades del CMS.

Includes	Este componte contiene un conjunto de ficheros indispensables para el funcionamiento de Drupal.
Misc	Incluye elementos que tienen que ver con el diseño, y funcionamiento (imágenes, ficheros js, etc.).
DataBase	Representa la base de datos.

Tabla 4.1 Descripción de los componentes.

4.4 Modelo de prueba.

En este caso se realiza la prueba de caja negra. Éstas consisten en pruebas aplicadas a la interfaz del sistema para probar su funcionalidad como un todo. No tienen que ver con el código interno del software, en ellas se definen las entradas, salidas y condiciones del software. Este tipo de prueba permite detectar funciones incorrectas o ausentes, como son los errores de interfaz, rendimiento, inicialización y terminación, en estructuras de datos o en acceso a las Bases de Datos externas.

Nombre del caso de uso: Autenticar usuario

ENTRADA	RESULTADOS	CONDICIONES
El usuario introduce nombre y/o contraseña incorrecta.	El sistema emite un mensaje indicando que no se reconoce el nombre de usuario o contraseña.	Se mantiene la pantalla de autenticación y no se le permite al usuario acceder a la aplicación.
El usuario no registrado trata de iniciar sesión pero deja algún campo del formulario vacío.	El sistema emite un mensaje indicando que los campos dejados vacíos son requeridos.	
El usuario se autentica correctamente.	El sistema autentica al usuario no registrado y determina en que nivel de acceso se encuentra.	

Nombre del caso de uso: Controlar Acceso a la Información.

Entrada	Resultados	Condiciones
El administrador quiere crear un rol pero no introduce el nombre del rol.	El sistema emite un mensaje de error “Debe indicar un nombre de rol válido”.	La operación se repite hasta que el administrador corrige el error.
El administrador quiere verificar las reglas de acceso pero deja el campo vacío.	El sistema emite un mensaje de error “No se ha introducido ningún valor. Introduzca una cadena de prueba y vuelva a intentarlo.”	La operación se repite hasta que el administrador corrige el error.

Nombre del caso de uso: Administrar Usuarios

Entrada	Resultados	Condiciones
El administrador desea crear un usuario pero no introduce los datos en los campos obligatorios (Nombre de usuario, Dirección de correo-e, Contraseña).	El sistema emite un mensaje de error “El campo (nombre del campo que dejó vacío) es necesario”.	La operación se repite hasta que el administrador corrige el error.
El administrador desea editar un usuario pero no introduce los datos en los campos obligatorios (Nombre de usuario, Dirección de correo-e).	El sistema emite un mensaje de error “El campo (nombre del campo que dejó vacío) es necesario”.	La operación se repite hasta que el administrador corrige el error.
El administrador desea buscar un usuario pero no introduce los datos en el campo.	El sistema emite un mensaje de error “Introduzca palabra clave”.	La operación se repite hasta que el administrador corrige el error.

Nombre del caso de uso: Gestionar la información de servicios web.

Entrada	Resultados	Condiciones
El publicador necesita gestionar la información de un proveedor, pero no introduce los datos que va a adicionar en los campos obligatorios (nombre, descripción, identificador, URL, relaciones)	El sistema emite un mensaje de error “El (los) campo(s) no pueden estar vacíos”.	La operación se repite hasta que el publicador corrige el error.
El publicador necesita gestionar la información de una especificación técnica, pero no introduce los datos que va a adicionar en los campos obligatorios (identificador, descripción).	El sistema emite un mensaje de error “El (los) campo(s) no pueden estar vacíos”.	La operación se repite hasta que el publicador corrige el error.

Nombre del caso de uso: Buscar

Entrada	Resultados	Condiciones
El usuario necesita hacer búsquedas de servicios, proveedores, especificaciones técnicas, pero no introduce los datos en los campos obligatorios.	El sistema emite un mensaje de error “El (los) campo(s) no pueden estar vacíos”.	La operación se repite hasta que el usuario corrige el error.

Nombre del caso de uso: Administrar Sistema

Entrada	Resultados	Condiciones
El administrador desea administrar la aplicación, pero no introduce los datos en los campos obligatorios en las diferentes acciones a realizar	El sistema emite un mensaje de error “El campo (nombre del campo que dejó vacío en la sección que se encuentra) es necesario”.	La operación se repite hasta que el administrador corrige el error.

(Gestión de contenido, Configuración del sitio, Construcción del sitio, Registros).		

4.5 Conclusiones

En este capítulo fue realizada la modelación de los nodos en los que será distribuida la aplicación, especificando para cada uno de éstos el protocolo de comunicación, mediante el diagrama de despliegue. Se representaron las dependencias entre los componentes software a través del diagrama de componentes y por último se realizó la descripción de los casos de prueba de integración de cada uno de los casos de uso del sistema.

CONCLUSIONES

Hoy en día el uso de los servicios web resulta muy ventajoso al alcanzar la interoperabilidad entre los sistemas conectados en la red, para esto es necesaria una colección de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Sin importar el lenguaje de programación sobre los cuales estas aplicaciones están desarrolladas, ni la plataforma sobre la que se ejecutan; siempre que tengan implementados servicios web para comunicarse, lo podrán hacer sin ninguna complicación.

En la universidad se hace necesario establecer un control de los servicios web que se realizan, y quien los realiza, es por eso que la arquitectura básica de servicios web debe incluir lo necesario para el intercambio de mensajes entre los proveedores, además de poner en conocimiento la existencia de esos servicios y la forma en que sean encontrados para poder utilizarlos.

Una vez estudiado, el protocolo UDDI, encargado de permitir el descubrimiento y la publicación de los servicios web se llegaron a las siguientes conclusiones:

• El trabajo alcanzó satisfactoriamente los objetivos propuestos: Diseño y montaje de un cliente para la administración de un registro UDDI (Descripción e Integración Universal del Descubrimiento) en software libre que cumpla con las especificaciones y estándares internacionales.

• Se realizaron pruebas para garantizar la seguridad y protección de los datos consecuente con el nivel de seguridad requerido.

• Se unificó la información referente al diseño, implementación, uso y estándares adaptados a la arquitectura SOA.

• Se demostró que la aplicación implementada tiene calidad por lo tanto se puede proceder con confianza a su utilización.

• Se realizaron pruebas para garantizar la seguridad y protección de los datos consecuente con el nivel de seguridad requerido.

• Se unificó la información referente al diseño, implementación, uso y estándares adaptados a la arquitectura SOA.

• Se demostró que la aplicación implementada tiene calidad por lo tanto se puede proceder con confianza a su utilización.

RECOMENDACIONES

Se recomienda:

- Extender el sistema de manera que pueda ser utilizado no sólo en la Universidad, sino en cualquier empresa que requiera de una UDDI.
- Que se ejecuten los requerimientos no funcionales de software con las particularidades necesarias.
- Continuar el desarrollo de este sistema, adicionándole nuevas funcionalidades, adecuándolo más a las demandas del creciente auge de servicios de la universidad y haciéndolo más útil y provechoso.
- Que el sistema aproveche la información de logs generada de forma centralizada en la aplicación, para poder controlar los movimientos de cada usuario dentro de éste.
- Que se pueda en un futuro adaptar a UDDI a un entorno más específico o centralizado.
- Se recomienda que el trabajo sea tomado como material de estudio para aquellas personas que vayan a realizar una aplicación similar.

REFERENCIAS BIBLIOGRAFICAS

[1].- Saffirio, Mario. ¿Qué son los Web Services?, Tecnologías de Información y Arquitectura de Sistemas, 2006. [En línea] [Citado el: 9 de marzo del 2008] Disponible en Web: **http://msaffirio.wordpress.com/2006/02/05/%C2%BFque-son-los-web-services/**.

[2].- Vázquez, Elisa Clavero y Sivila, Dianelis Tur. *Diseño de un catálogo de servicios Web basado en las especificaciones y normas de la UDDI para la plataforma de informatización en la UCI.* UCI. La Habana: s.n., 2007. pág. 156, Tesis de diplomado.

[3].- Aguilar, Ronny Zamora y Ramos, Alberto Tamayo. *Sistema para la informatización del Convenio Cuba-Venezuela. Módulo ficha mixta.* UCI. La Habana: s.n., 2007. pág. 137, Tesis de diplomado.

[4].- Cuerda García Xavier y Minguillón Alfonso Julià. *Introducción a los Sistemas de Gestión de Contenidos (CMS) de código abierto*, 2008. [En línea] [Citado el: 9 de marzo del 2008] [Disponible en: **http://mosaic.uoc.edu/articulos/cms1204.html]**.

[5].- ANDRAM. *CMS gratuitos/de código abierto*, 2008. [En línea] [Citado el: 9 de marzo del 2008] [Disponible en: **http://andram.nireblog.com/cat/cms]**.

[6].- Wikimedia Foundation, Inc. *Drupal*, 2008. [En línea] [Citado el: 9 de marzo del 2008] [Disponible en: http://es.wikipedia.org/wiki/Drupal].

[7].- L. Welicki. *Patrones y Antipatrones*: una Introducción –Parte II.

[8].- Lockhart, Thomas. *Tutorial de PostgreSQL, El equipo de desarrollo de PostgreSQL*. [En línea] [Citado el: 10 de marzo del 2008] Disponible en Web: http://palomo.usach.cl/Docs/postgres/Postgres-Tutorial.pdf.

[9].- Jacaboson, I., Booch, G., Rumbaugh J. *El Proceso Unificado de Desarrollo de Software*, 2000 Addison Wesley.

[10].- Schmuller, Joseph. *Aprendiendo UML en 24 horas*. 2000. pág. 425. Vol. 1.

[11].- Palmero, Julio Ruiz. *Manual de NVU.* [En línea] [Citado el: 10 de marzo del 2008] Disponible en Web: http://www.iestorredelprado.net/nvu/nvu/MANUAL%20DE%20NVU.%20cap%201.pdf.

BIBLIOGRAFIA

[1].- Aguilar, Ronny Zamora y Ramos, Alberto Tamayo. *Sistema para la informatización del Convenio Cuba-Venezuela. Módulo ficha mixta.* UCI. La Habana: s.n., 2007. pág. 137, Tesis de diplomado.

[2].- ANDRAM. *CMS gratuitos/de código abierto*, 2008. [En línea] [Citado el: 9 de marzo del 2008] [Disponible en: **http://andram.nireblog.com/cat/cms]**

[3].- Cuerda García Xavier y Minguillón Alfonso Julià. *Introducción a los Sistemas de Gestión de Contenidos (CMS) de código abierto*, 2008. [En línea] [Citado el: 9 de marzo del 2008] [Disponible en: **http://mosaic.uoc.edu/articulos/cms1204.html]**

[4].- Jacaboson, I., Booch, G., Rumbaugh J. *El Proceso Unificado de Desarrollo de Software*, 2000 Addison Wesley

[5].- L. Welicki. *Patrones y Antipatrones*: una Introducción –Parte II.

[6].- Lockhart, Thomas. *Tutorial de PostgreSQL, El equipo de desarrollo de PostgreSQL*. [En línea] [Citado el: 10 de marzo del 2008] Disponible en Web: http://palomo.usach.cl/Docs/postgres/Postgres-Tutorial.pdf

[7].- Palmero, Julio Ruiz. *Manual de NVU.* [En línea] [Citado el: 10 de marzo del 2008] Disponible en Web: http://www.iestorredelprado.net/nvu/nvu/MANUAL%20DE%20NVU.%20cap%201.pdf.

[8].- *Saffirio, Mario*. ¿Qué son los Web Services?, *Tecnologías de Información y Arquitectura de Sistemas, 2006. [En línea] [Citado el: 9 de marzo del 2008] Disponible en Web:* ***http://msaffirio.wordpress.com/2006/02/05/%C2%BFque-son-los-web-services/***

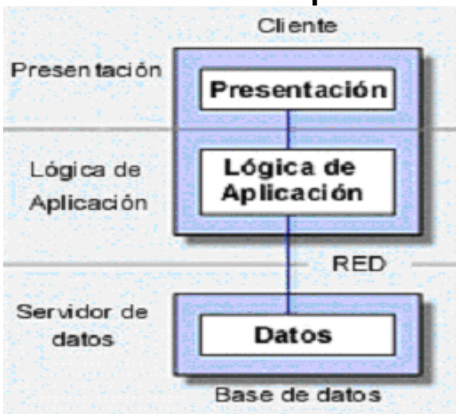
[9].- Schmuller, Joseph. *Aprendiendo UML en 24 horas*. 2000. pág. 425. Vol. 1.

[10].- Vázquez, Elisa Clavero y Sivila, Dianelis Tur. *Diseño de un catálogo de servicios Web basado en las especificaciones y normas de la UDDI para la plataforma de informatización en la UCI.* UCI. La Habana: s.n., 2007. pág. 156, Tesis de diplomado.

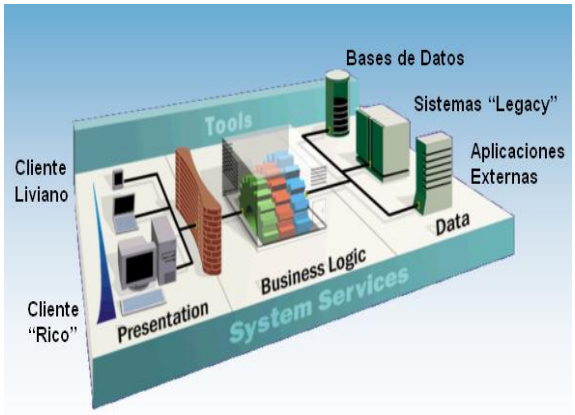
[11]. – Wikimedia Foundation, Inc. *Drupal*, 2008. [En línea] [Citado el: 9 de marzo del 2008] [Disponible en: http://es.wikipedia.org/wiki/Drupal]

ANEXOS

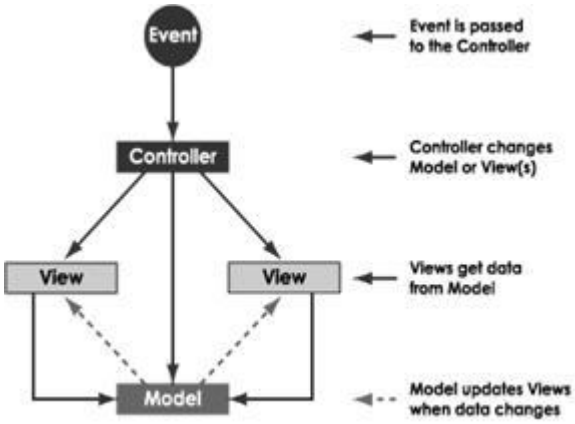
Anexo1. Modelo Cliente – Servidor de dos capas.



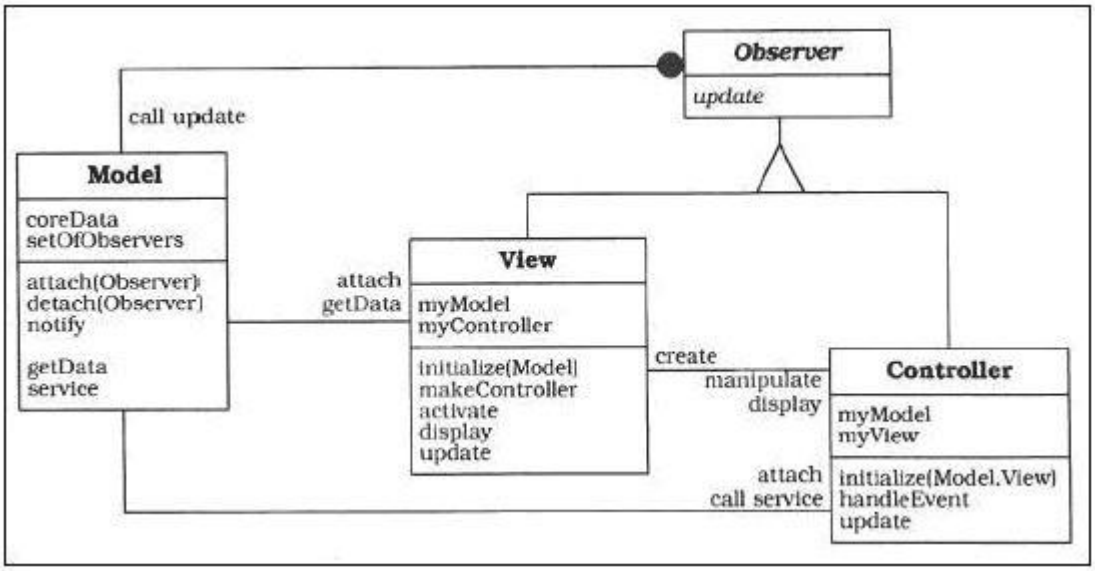
Anexo 2: Modelo Cliente – Servidor de tres capas.



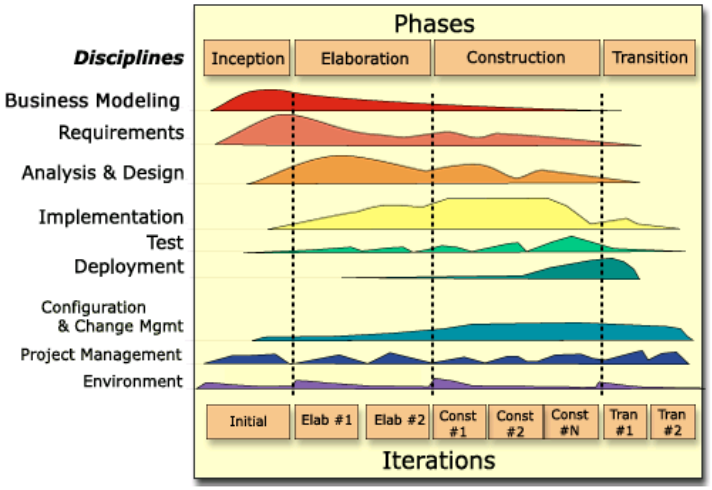
Anexo 3: Funcionamiento del patrón MVC.



Anexo 4: Estructura del patrón MVC.



Anexo 5: Flujos de trabajo de RUP.



Flujos de trabajo:

- **Modelamiento del negocio.**
- **Requerimientos.**
- **Análisis y diseño.**
- **Implementación.**
- **Prueba (Testeo).**
- **Instalación.**
- **Administración del proyecto.**
- **Administración de configuración y cambios.**
- **Ambiente.**

GLOSARIO DE TÉRMINOS Y SIGLAS

Apache: Es un servidor HTTP de código abierto para plataformas Unix (BSD, GNU/Linux, etcétera), Windows y otras, que implementa el protocolo HTTP/1.1.

ASP: Es una tecnología del lado servidor de Microsoft para páginas web generadas dinámicamente, que ha sido comercializada como un anexo a Internet Information Server (IIS).

BSD: Es utilizado para identificar un sistema operativo derivado del sistema Unix.

C: Es un lenguaje de programación creado en 1972 por Ken Thompson y Dennis M. Ritchie en los Laboratorios Bell como evolución del anterior lenguaje B, a su vez basado en BCPL.

CGI: Es una importante tecnología de la World Wide Web que permite a un cliente (explorador web) solicitar datos de un programa ejecutado en un servidor web. CGI especifica un estándar para transferir datos entre el cliente y el programa.

CodeIgniter: Es un entorno de trabajo web para el desarrollo de aplicaciones web en PHP que facilita la escritura de código fuente en este lenguaje y reduce el tiempo necesario para lograrlo. Está basado en el sistema modelo vista controlador.

Framework: Es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, un framework puede incluir soporte de programas, bibliotecas y un lenguaje de scripting entre otros softwares para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

GNU: Es un acrónimo recursivo que significa GNU No es Unix (*GNU is Not Unix*). Puesto que en inglés "*gnu*" (en español "ñu") se pronuncia igual que "*new*", Richard Stallman recomienda pronunciarlo "*guh-noo*". En español, se recomienda pronunciarlo ñu como el antílope africano o fonéticamente; por ello, el término mayoritariamente se deletrea (G-N-U).

GPL: Es la Licencia pública general de GNU.

GTK+: (The GIMP Toolkit). Es un grupo importante de librerías o rutinas para desarrollar interfaces gráficas de usuario (GUI) para entornos gráficos GNOME, XFCE y ROX de Sistemas Linux.

HTTP: Es el protocolo de Transferencia de Hipertextos. Modo de comunicación para solicitar páginas web.

HTML: Es un Lenguaje usado para escribir documentos para servidores World Wide Web. Es una aplicación de la ISO Standard 8879:1986. Es un lenguaje de marcas. Los lenguajes de marcas no son equivalentes a los lenguajes de programación aunque se definan igualmente como "lenguajes". Son sistemas complejos de descripción de información, normalmente documentos, que se pueden controlar desde cualquier editor ASCII.

IDE: (Integrated Development Environment, en español: Entorno de Desarrollo Integrado). Es un programa compuesto por un conjunto de herramientas para un programador. Puede dedicarse en exclusiva a un solo lenguaje de programación o bien, poder utilizarse para varios.

Interfaz: Es el de mediación, entre hombre y máquina. La interfaz es lo que "media", lo que facilita la comunicación, la interacción, entre dos sistemas de diferente naturaleza, típicamente el ser humano y una máquina como el computador. Esto implica, además, que se trata de un sistema de traducción, ya que los dos "hablan" lenguajes diferentes: verbo-icónico en el caso del hombre y binario en el caso del procesador electrónico.

Internet: Es un método de interconexión descentralizada de redes de computadoras implementado en un conjunto de protocolos denominado TCP/IP y garantiza que redes físicas heterogéneas funcionen como una red lógica única, de alcance mundial.

JSP: Es la tecnología para generar páginas web de forma dinámica en el servidor, desarrollado por Sun Microsystems, basado en scripts que utilizan una variante del lenguaje java. La tecnología JSP, o de JavaServer Pages, es una tecnología Java que permite a los programadores generar dinámicamente HTML, XML o algún otro tipo de página web. Esta tecnología permite al código Java y a algunas acciones predefinidas ser embebidas en el contenido estático.

Linux: Es la denominación de un sistema operativo tipo Unix (también conocido como GNU/Linux) y el nombre de un núcleo. Es software libre y del desarrollo del código abierto, cuyo código fuente está disponible públicamente, para que cualquier persona pueda libremente usarlo, estudiarlo, redistribuirlo,

comercializarlo y, con los conocimientos informáticos adecuados, modificarlo.

Mac OS X: Es un sistema operativo basado en UNIX, desarrollado íntegramente por Apple para los ordenadores Macintosh.

Microsoft: Es la compañía que manufactura los sistemas de operación DOS y Windows.

Meta-Datos: Datos que sirven para describir algo, por ejemplo, puede ser un registro que almacene información de un dato dado.

OASIS: acrónimo de (Organization for the Advancement of Structured Information Standards) es un consorcio internacional sin fines de lucro que orienta el desarrollo, la convergencia y la adopción de los estándares e-business.

Open Source: (Código abierto). Es el término con el que se conoce al software distribuido y desarrollado libremente.

Página web: Es una fuente de información adaptada para la World Wide Web y accesible mediante un navegador de Internet.

Perl: Es un lenguaje de programación desarrollado por Larry Wall inspirado en otras herramientas de UNIX como son: sed, grep, awk, c-shell.

PHP: Es un ambiente script del lado del servidor que permite crear y ejecutar aplicaciones web dinámicas e interactivas. Con PHP se pueden combinar páginas HTML y scripts. Con el objetivo de crear aplicaciones potentes.

PostgreSQL: Es un motor de base de datos, es servidor de base de datos relacional libre, liberado bajo la licencia BSD.

Procedimiento: Es la secuencia de acciones concatenadas entre sí, que ordenadas en forma lógica permite cumplir un fin u objetivo predeterminado.

Ruby: Es un lenguaje de programación reflexivo y orientado a objetos (lenguaje interpretado).

Ruby on Rails: También conocido como RoR o Rails es un framework de aplicaciones web de código abierto escrito en el lenguaje de programación Ruby, siguiendo el paradigma de la arquitectura Modelo Vista Controlador.

Registro: Estructura compleja que define UDDI para descubrir e interactuar con los Servicios web. Esta estructura está compuesta por cuatro tipos de datos básicos para la información del negocio y del servicio.

Registro UDDI: Es el conjunto de nodos.

RPC: *(Remote Procedure Call).* Conjunto de herramientas software desarrolladas por un consorcio de fabricantes y diseñadas para asistir a los diseñadores en la creación de aplicaciones distribuidas.

SOA: La Arquitectura Orientada a Servicios (en inglés *Service-Oriented Architecture*), es un concepto de arquitectura de software que define la utilización de servicios para dar soporte a los requerimientos de software del usuario.

SOAP: (Simple Object Access Protocol, Protocolo de Acceso simple a objetos). Especificación XML para la formación de los mensajes intercambiados entre los sistemas distribuidos y la red. Es un protocolo estándar creado por Microsoft, IBM y otros, está actualmente bajo el auspicio de la W3C que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML. SOAP es uno de los protocolos utilizados en los servicios web.

Scaffolding o **Scaffold:** Es una palabra que está en inglés y en español significa Andamio, pero en programación el scaffolding es un método para construir aplicaciones basadas en bases de datos.

Software libre: Es la denominación del software que brinda libertad a los usuarios sobre su producto adquirido y por tanto, una vez obtenido, puede ser usado, copiado, estudiado, modificado y redistribuido libremente.

SQL: Es un lenguaje declarativo de acceso a bases de datos que permite especificar diversos tipos de operaciones sobre las mismas. Aúna características del álgebra y el cálculo relacional permitiendo lanzar consultas con el fin de recuperar información de interés de una base de datos.

Servicio web: Es una colección de protocolos y estándares que sirven para intercambiar datos entre aplicaciones.

Sistema Taxonómicos: Son sistemas creados para la clasificación y categorización de ciertos datos o informaciones.

TCP/IP: (*Transmission Control Protocol/Internet protocol).* Sistema de protocolos, definidos en RFC 793, en los que se basa buena parte de Internet. El primero se encarga de dividir la información en paquetes en origen, para luego recomponerla en destino, mientras que el segundo se responsabiliza de dirigirla adecuadamente a través de la red.

UNIX: Es un sistema operativo multitarea, multiusuario. Gran parte de las características de otros sistemas más conocidos como MS-DOS están basadas en este sistema muy extendido para grandes servidores. Internet no se puede comprender en su totalidad sin conocer el Unix, ya que las comunicaciones son una parte fundamental en Unix.

UDDI: (Universal Description, Discovery and Integration, Descripción, Descubrimiento e Integración). Es un elemento básico sobre el que se asientan los web services, hace posible que empresas pueden tanto publicar como encontrar servicios web. UDDI provee un mecanismo para que los negocios se "describan" a si mismos y los tipos de servicios que proporcionan y luego se pueden registrar y publicarse en un Registro UDDI. Tales negocios publicados pueden ser buscados, consultados o "descubiertos" por otros negocios utilizando mensajes con SOAP.

W3C: (*World Wide Web Consortium).* Organización apadrinada por el MIT y el CERN, entre otros, cuyo cometido es el establecimiento de los estándares (especificaciones, directrices, software, y herramientas) relacionados con WWW.

WWW, WEB o W3: *World Wide Web*. *Telaraña mundial* es un sistema de documentos de hipertexto accesibles a través de Internet. Podríamos decir estrictamente que la WEB es la parte de Internet a la que accedemos a través del protocolo HTTP mediante un navegador web.

WYSIWYG: Es el acrónimo de What You See Is What You Get (en inglés, "lo que ves es lo que obtienes").

WSDL: (Web Service Definition Language, Lenguaje de descripción de servicios web). Especificación XML para la formación del documento de descripción de un servicio web. Identifica los métodos, funciones y parámetros necesarios para invocar un determinado servicio. Así, un usuario puede crear una aplicación cliente que comunica con el servicio web.

XML: son las siglas en inglés de eXtensible Markup Language (lenguaje de marcado ampliable o extensible) desarrollado por el World Wide Web Consortium (W3C). Es un estándar para describir datos y crear etiquetas. Las características especiales son la independencia de datos, o de la separación de los contenidos de su presentación. Es un metalenguaje que permite diseñar un lenguaje propio de etiquetas para múltiples clases de documentos. Los documentos XML se componen de unidades de almacenamiento llamadas entidades (*entities*), que contienen datos analizados (*parsed*) o sin analizar (*unparsed*).

XML-RPC: Es un protocolo de llamada a procedimiento remoto que usa XML para codificar los datos.

Zend: Es una compañía encargada del desarrollo del popular lenguaje de programación PHP. La compañía Zend Technologies, fundada por los creadores de PHP.