UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

DISEÑO DE LA INTRANET DE LAS RESIDENCIAS DE PROTOCOLO DEL CONSEJO DE ESTADO DE LA REPÚBLICA DE CUBA

Trabajo de diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autores: Dayelis Blanco Hernández

Máxora Rorayma Castro Pérez

Tutor: Lic. Luis Guzmán Hernández

Ciudad de la Habana, Junio 2007

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 año _____.

_____ _____

Dayelis Blanco Hernández

Autor

Máxora Rorayma Castro Pérez

Autor

_____

Luis Guzmán Hernández

Tutor

Si quieres construir un barco, no empieces por buscar madera, cortar tablas o distribuir el trabajo, sino que primero debes evocar en los hombres el deseo del mar libre y ancho.

Antoine de Saint-Exupéry

Dedicatoria

A nuestros padres por entregarnos todo
el amor y la confianza que necesitamos para alcanzar un sueño.

Agradecimientos

A Fidel Castro y a los mártires de la Revolución por entregar la vida a la patria.
A nuestros padres porque todo lo que somos es el fruto de su sacrificio y amor infinito.
A Luis por su amistad y por su empeño en enseñarnos.
A nuestros hermanos Dayanis y Camilo por acompañarnos en todos los momentos.
A nuestros amigos Laritza, Yaneika, Yasnay y Carlos por las alegrías, las tristezas y el tiempo compartido.
A la Universidad de Ciencias Informáticas por abrirnos las puertas de la sabiduría y la verdad.

Máxora
A mi abuela Coca por dedicarme la vida y por ser la persona mas especial que conoceré jamás.
A mi abuela Fina por su ternura y ejemplo.
A mi prima Cory por enseñarme los colores de la vida con sus dulces locuras.

Dayelis
A mi primer pensamiento de todos los días, que llena mi vida de magia, sueños y auténtico amor, Osiel.
A Elenis, Roberto, Camejo y Jose por su extraordinaria amistad.
A mi amiga Marisley por mantenerse a mi lado a pesar de la distancia.
A Ledián por apoyarme con agrado y dedicación.

A todos los que alguna vez a lo largo del camino nos tendieron afectuosamente la mano.

Resumen

Las Residencias de Protocolo de Consejo de Estado de la República de Cuba es una organización encargada de la importante misión de brindar alojamiento en sus instalaciones a las visitas de mayor relevancia que se reciben en el país. Teniendo en cuenta el aporte que brindan las aplicaciones Web para la prestación de servicios online y la optimización con rapidez de la información, unificándola y facilitando su tratamiento, el departamento de informática de dicha organización ha concebido la creación de una Intranet que opere sobre su red interna y que sea la bienvenida digital para los huéspedes, inmediatamente que estos accedan al navegador.

En el presente trabajo de diploma se realiza el diseño de la Intranet para las Residencias de Protocolo que brindará a los huéspedes información actualizada sobre los servicios que tiene a su disposición y sobre la organización. Los contenidos de la Intranet serán gestionados en un módulo administrativo y los encargados de realizar la gestión serán los usuarios clasificados como administradores del sistema.

El diseño de la aplicación queda conformado según la realización de los flujos de trabajo de la metodología RUP para la ingeniería de software: Modelo de negocio, Requerimientos, Análisis y diseño.

Índice

Introducción

La Dirección de Residencias de Protocolo del Consejo de Estado de la República de Cuba fue creada por la Resolución No. 733 del 13 de enero de 1993. Actualmente esta institución cuenta con 57 residencias, de ellas 47 en Ciudad de La Habana, 9 en la playa de Varadero y una en la playa Santa María del Mar. Además cuenta con tres salones de protocolo, de ellos dos en el aeropuerto “José Martí”, un centro de elaboración de alimentos, un complejo de tintorería lavandería, un taller de mecánica, un área de mantenimiento, instalaciones de almacenamiento y varias oficinas administrativas.

Desde su fundación las Residencias de Protocolo han constituido el lugar por excelencia para hospedar de manera gratuita a las visitas más importantes que se reciben en el país, como Jefes de Estado, Primeras Damas, Embajadores, delegaciones internacionales, grandes personalidades de la ciencia y la cultura e invitados de la máxima dirección del país, que llegan a Cuba desde cualquier lugar del mundo de manera oficial o extraoficial con el fin de participar en actividades estratégicas y de relevancia como firma de convenios, cumbres, recepciones, coloquios, eventos de solidaridad, celebraciones y acciones de corte diplomático.

Cada una de las residencias está condicionada para brindar servicios de alta calidad donde se incluyen alojamiento, gastronomía, paseos por lugares históricos, reparación instantánea de averías, transportación y lavandería. Las residencias están decoradas con obras de artes originales, jardinería de primera y todo el personal que labora para la atención a los huéspedes posee una excelente calificación.

La efervescencia mundial de las nuevas tecnologías de la informática y las comunicaciones (NTIC), ha marcado pautas y redefinido estructuras en prácticamente todas las esferas de la sociedad, constituyen un elemento que aporta eficiencia, calidad y competencia a todo proyecto y prescindir de ellas en la actualidad significa truncar el desarrollo. Específicamente en el campo de las redes computacionales y la Web las aplicaciones de las NTIC se incrementan a diario y crecen impetuosamente las entidades que desean brindar servicios a través de la red o simplemente contar con un sitio que la haga presente frente al enorme tráfico que circula por esta autopista virtual.

Por otra parte el manejo estratégico de la información es uno de los temas más investigados y desarrollados actualmente en el mundo, pues a partir de aquí se generan los

principales flujos de trabajo y se toman decisiones que afectan los procesos y las funciones futuras de toda organización.

La información en las empresas es tan importante como lo son los recursos humanos o la materia prima, pues sin la información adecuada, la empresa estará en desventaja con respecto a sus similares. Si no permanece a la vanguardia en la tecnología de la información que va apareciendo día a día, la entidad carecería de procesos innovadores y actualizados. Otro factor de suma importancia a tener en cuenta es el hecho de que la información llegue cuanto antes y de manera fiable a los directivos, organizadores y otros usuarios de interés para la entidad.

Una de las herramientas innovadoras en este contexto es lo que se conoce como Intranet o red de comunicación dentro de una organización. En general, una Intranet es una red privada de ordenadores basada en los estándares de Internet que sólo determinados miembros de una organización pueden ver.

“Las compañías pueden crear por tanto, dentro de sus paredes informáticas, una versión segura y administrable de la “World Wide Web”. Estas Webs internas están creciendo debido a la explosión en el uso y entendimiento de la tecnología de Internet. Mediante ellas, se pueden intercomunicar todos los miembros de la empresa y trasmitir el contenido necesario para lograr un mejor desempeño de sus funciones y procesos.” [1]

Las principales ventajas que se obtienen al utilizar la Intranet son: facilidad para compartir información y archivos, utilización del correo electrónico y otros medios de comunicación, ahorro de costos a nivel de recursos (papel, tiempo, etc.) y la obtención de un mejor servicio de comunicación.

Las Intranets aportan un valor sin precedentes a la distribución de la información, la automatización de los grupos de trabajo y el acceso a la información corporativa.

Por todo ello es que las Residencias de Protocolo cuentan con una red interna y han decidido crear una Intranet que se convierta en la cara digital del recinto para los huéspedes que en este caso serán sus principales usuarios, por lo que surge la siguiente problemática: Diseñar una Intranet para el servicio de protocolo del Consejo de Estado de la República de Cuba.

En la actualidad las Residencias de Protocolo no aprovechan las ventajas de las aplicaciones Web para publicar información digital especializada y distribuirla en la red. La organización estaría mejor preparada para operar y manejar los nuevos retos que generan los

altos volúmenes de conocimientos y de información disponible, si contara con un soporte digital que le diera la capacidad de almacenar, catalogar y hacer disponible en forma oportuna y a costos razonables los contenidos de los que desee nutrir a los huéspedes. Además desperdician la oportunidad del intercambio interactivo con los usuarios finales que posibilite satisfacer sus demandas y solicitudes. Por otra parte no contar con una herramienta soportada por las aplicaciones Web, la deja en desventaja con otras organizaciones similares y le resta seriedad y el profesionalismo al trabajo que realizan.

Para solucionar la problemática existente se define el **objeto de estudio:** las aplicaciones Web como herramientas para la gestión empresarial.

Se ha delimitado como **campo de acción** la Intranet como medio para compartir recursos informativos eficientemente y adaptarlos a las necesidades de los usuarios en el Servicio de Protocolo del Consejo de Estado de la República de Cuba.

El **objetivo principal** del presente trabajo de diploma consiste en:

Realizar los flujos de trabajos de la ingeniería de software (Modelo de negocio, Requerimientos y Análisis y diseño) para la creación de una Intranet para el Servicio de Protocolo del Consejo de Estado de la República de Cuba.

En vista a dar cumplimiento a este objetivo se plantean las siguientes **preguntas científicas**:

¿Qué ventajas tiene el diseño de una Intranet para los Servicios de Protocolo del Consejo de Estado de la República de Cuba?

¿Qué condiciones existen para el diseño de la Intranet en el Servicio de Protocolo del Consejo de Estado de la República de Cuba?

¿Cómo proceder para el diseño de la Intranet del Servicio de Protocolo del Consejo de Estado de la República de Cuba?

Para dar respuesta a las preguntas científicas se han trazado las siguientes **tareas**:

1. Estudiar el entorno de trabajo para capturar los requisitos que debe cumplir el sistema.
2. Investigar el estado actual de las tecnologías que se utilizan para el desarrollo de aplicaciones como la que se pretende crear.
3. Seleccionar la metodología para modelar la Intranet.
4. Seleccionar las herramientas necesarias para el desarrollo del sistema.
5. Elaborar el análisis y diseño del la aplicación.

Se pretende finalmente conseguir un producto de software que resuelva las necesidades del recinto en cuanto a cohesión de estructura, flujo de información interna y que además represente los esfuerzos de la organización por brindar una atención de excelencia.

El presente trabajo de diploma está estructurado de la siguiente manera:

El primer capítulo contiene la fundamentación teórica, en el que se hace un análisis de las tendencias actuales de sistemas similares y un estudio crítico-valorativo de las disímiles tecnologías, metodologías y software más usados en la actualidad.

El segundo capítulo se relaciona con las características del sistema: Definición del problema, descripción de los procesos que serán objeto de automatización, modelación del negocio y definición de los requerimientos.

El tercer capítulo contiene el análisis y diseño del sistema de manera que responda y de cumplimiento a los requerimientos definidos en la modelación del negocio.

Capítulo I
Fundamentación Teórica

Para realizar la Intranet de las Residencias de Protocolo del Consejo de Estado se hizo un análisis de aplicaciones Web similares existentes dentro y fuera del país, con el objetivo de encontrar aspectos novedosos y funcionalidades útiles que pudieran ser incluidas en el proyecto. También se realizó un estudio de las tendencias y aplicaciones que tienen en la actualidad las herramientas, metodologías y tecnologías que pueden ser utilizadas para elaborar el sistema.

1.1 Tendencias Actuales.

Las aplicaciones Web son un sistema informático que los usuarios utilizan accediendo a un servidor Web a través de Internet o de una Intranet que generan dinámicamente una serie de páginas en un formato estándar como HTML o XHTML, soportados por navegadores Web comunes, además pudiera funcionar como cualquier conjunto de páginas Web que interactúen con el usuario ofreciéndole la información solicitada y recogiendo datos del mismo. El desarrollo de una aplicación Web permite publicar un catálogo electrónico de productos, manejo de inventarios, órdenes de compra, publicación de información con acceso restringido a ciertos usuarios, actualización y mantenimiento de sitio Web y en general, permite publicar cualquier tipo de información que se pueda incorporar a una base de datos.

Las aplicaciones Web son populares debido a la practicidad del navegador Web y la habilidad para actualizarlas y mantenerlas sin distribuir e instalar software en miles de clientes potenciales.

Algunas de las más usadas son los sitios Web, los portales e Intranets, destacándose las wikis, weblogs y las tiendas en línea.

En la actualidad las aplicaciones Web que representan a entidades dedicadas al hospedaje como hoteles o residencias, exponen con riqueza de detalles toda la información que el visitante necesita conocer.

En estos sitios se muestran fotos y descripciones de los espacios de interés que posea el recinto como son las habitaciones, centros recreativos y de ocio, lugares

históricos, salones para la realización de eventos empresariales, negociaciones, bodas, entre otros.

También como tendencia suelen dedicarle secciones a los principales servicios que se brindan. En la sección de alojamiento informan la tarifa de reservado según la categoría de la habitación. En la de gastronomía y coctelería exponen un compendio de los restaurantes, bares y de las ofertas que distinguen al lugar. Para los servicios de esparcimiento y recreación muestran un recuento de los espectáculos, juegos, instalaciones, paseos y otras opciones que se ofrecen. Además permiten realizar la reservación de manera online, solicitando al usuario que introduzca la fecha de llegada, la fecha de salida, el tipo de habitación, datos personales y demás información necesaria para el hospedaje. Es usual que las aplicaciones sean accesibles en varios idiomas fundamentalmente el inglés y que ofrezcan una dirección de email para que la organización sea contactada por los interesados.

En Cuba desde hace varios años y por causa del inicio del periodo especial el turismo se convirtió en una parte fundamental de nuestra economía. Existen varios recintos hoteleros a lo largo de todo el archipiélago que contribuyen en gran medida al aumento de los ingresos monetarios del país, es por ello que se han publicado varias aplicaciones Web como una vía para promocionar nuestras instalaciones en el mundo entero.

Sol Meliá es una de las mayores cadenas hotelera que ha invertido en Cuba y cuenta con un portal en Internet que recoge una lista de vínculos a páginas dedicadas a exponer con abundantes fotos e información cada una de sus instalaciones. La mayoría de los hoteles nacionales e internacionales utilizan ese estilo, es decir, no poseen un sitio oficial, sino que su presencia en la Web es a través del sitio de la cadena a la que pertenecen.

Después del análisis de aplicaciones Web similares se obtienen elementos que servirán de apoyo para el desarrollo de la Intranet, la cual se pretende que esté visible inmediatamente que los huéspedes accedan al navegador y será un elemento más que imprima experiencia, seriedad y solidez a la atención que los anfitriones desean brindar.

En dicha Intranet el huésped podrá ver toda la información referente a la historia de las residencias, incluyendo el sitial histórico y la figura de Celia Sánchez Manduley, protagonista de la fundación del complejo. Encontrará además detalles sobre las obras de arte que decoran su residencia y una lista de vínculos a sitios cubanos de interés

donde podrá enriquecer sus conocimientos sobre los logros de la revolución cubana, específicamente en el campo de la salud, la educación, la cultura, la ciencia y tendrá también el acceso a las ediciones digitales de nuestros principales órganos de prensa.

En cuanto a los servicios que se brindan por la entidad: reparación de averías y desperfectos, alojamiento, transporte y televisión por cable, el huésped podrá conocer mediante la Intranet los pormenores de cada uno de ellos. En el caso específico del servicio de lavandería podrá realizar su solicitud de manera online y en cuanto a la gastronomía se le informará el menú semanal del almuerzo y la cena así como los vinos, las bebidas y cócteles que tienen a su disposición. Tendrán acceso también a una galería de fotos que ilustrará tanto las residencias (interiores y exteriores) como dichos servicios. Además podrá emitir su grado de satisfacción mediante encuestas disponibles en la Intranet.

La aplicación estará diseñada para poder editar más versiones de la misma en diferentes idiomas. Además contará con un módulo administrativo a través del cual y mediante controles de acceso adecuados, se podrá actualizar toda la información contenida en la base de datos.

La organización posee un departamento de informática encargado de proveer los servicios de informática. En cada residencia existe una computadora con acceso permanente a Internet y a la red interna de la entidad.

1.2 Intranet.

Una Intranet es una red de área local (LAN) que tiene como base el protocolo TCP/IP de Internet y utiliza un sistema firewall que no permite acceder a la misma desde el exterior.

“El principal beneficio de la Intranet es que es una herramienta efectiva para combatir la pérdida de tiempo, esfuerzo y materiales dentro de una organización, al tiempo que genera nuevas oportunidades para la colaboración y productividad.

La Intranet provee a la organización de:

Ahorro: Elimina documentos, formularios, manuales, notas internas y demás documentación que obliguen al uso de papel.

Calidad: La red global está cargada de información, pero el tiempo que se debe emplear para su localización es a menudo demasiado para ofrecer soluciones rápidas a

las dificultades diarias. En la Intranet el usuario encuentra lo que necesita ya que la información se presenta mucho más organizada y seleccionada.

Comunicación: El sistema de mensajería implementado en una Intranet puede ofrecer comunicaciones internas y externas, en tiempo real y diferido.

Control: Los datos importantes de la organización no estarán tan sólo en la mesa de un directivo o en la de un mando intermedio. La información en una Intranet se puede modificar y consultar en tiempo real por aquellos que tengan los permisos.

Colaboración: Permite aprovechar la experiencia intelectual individual de los miembros de la organización.

Efectividad: Permite que los departamentos mantengan su propia documentación así como los datos que le pertenecen.

Eficiencia: Una Intranet elimina en porcentajes muy altos los métodos de trabajo tradicionales y en ocasiones poco eficientes como el teléfono, el fax o las reuniones.

Facilidad: Con la simple habilidad de saber manejar un navegador, cualquier usuario podrá trabajar con una Intranet.

Rendimiento: Todo el material que se puede tener de manera impresa en un catálogo, manual o libro, se puede implementar en una Intranet.” [1]

1.3 Sitio Web.

Un Sitio Web es un conjunto de archivos electrónicos y paginas Web referentes a un tema en particular, que incluye una página inicial de bienvenida, generalmente denominada home page, con un nombre de dominio y con una dirección específica.

Una página Web es un documento en su mayoría HTML/XHTML accesible generalmente mediante el protocolo HTTP de Internet.

Los sitios Web son empleados por las instituciones públicas y privadas, organizaciones e individuos para comunicarse con el mundo entero. Hoy en día, hay más de 100 millones de sitios Web en el mundo con dominios registrados.

Pueden ser de diversos géneros, destacando los sitios de negocios, servicio, comercio electrónico en línea, imagen corporativa, entretenimiento y sitios informativos.

Pero básicamente se dividen en dos grandes grupos, los sitios Web estáticos y los dinámicos.

"Un sitio Web dinámico es aquel en el que los usuarios pueden actuar directamente sobre el contenido del sitio o con otros usuarios del mismo. Hasta cierto punto, todos los sitios presentan algún tipo de interactividad, en el sentido de que los usuarios pueden elegir cómo explorar su contenido. Los sitios verdaderamente interactivos permiten a los usuarios manipular el contenido y, en algunos casos, incluso añadir su propio contenido. Un sitio que permite a un usuario incluir cuestiones de soporte técnico para otros usuarios se considera que es interactivo, mientras que un sitio en el que sólo se permite a los usuarios consultar respuestas preexistentes se considera que es estático.

Un sitio Web estático es aquel cuyo contenido es relativamente fijo, en el que los usuarios no pueden modificar ni el aspecto ni el ámbito de los datos que observan. En resumen, el visitante no tiene posibilidad de interactuar con el contenido del sitio, excepto en la elección del orden en que desea verlo. Cuando el usuario accede a un sitio estático puede elegir entre avanzar o retroceder por sus páginas y leer los artículos en cualquier orden, pero la presentación es relativamente rígida y no existe posibilidad de hacer nada con el contenido, excepto leerlo. Igual que sucede con la impresión en papel, una vez terminado el sitio estático, no cambia con el tiempo y los usuarios no pueden modificar ni su aspecto ni la forma en que actúa. Sin embargo, la mayoría de los sitios no son completamente estáticos; con el tiempo, se realizan cambios en sus páginas." [2]

1.4 Herramientas para el Desarrollo de la Aplicación Web.

Para la selección de las herramientas y metodologías que se usarán en el desarrollo del proyecto se indagó en diferentes fuentes bibliográficas sobre las ventajas que reportan, los resultados que ya se han obtenido con ellas y las potencialidades que ofrece.

1.4.1 Lenguaje de Representación Visual para el Modelado del Sistema.

UML (Unified Modeling Language)

UML es un lenguaje de representación visual que se ha diseñado realizando combinaciones de una gran cantidad de estándares, mas bien regido a través de tres metodologías, y que gracias a la iniciativa de sus tres creadores J. Rumbaugh, G. Booch e I. Jacobson, UML tiene un diseño completo desde sus inicios. Ha sido seleccionado

como leguaje de representación visual para el diseño de la Intranet debido a que permite modelar sistemas con tecnología orientada a objetos para visualizar, especificar, construir y documentar los artefactos de un sistema que involucra una gran cantidad de software.

Por otra parte, está compuesto por diversos elementos gráficos que se combinan para conformar diagramas, con la finalidad de presentar diversas perspectivas de un sistema, con el objetivo además de describirlos con cierto grado de formalismo, para que puedan ser entendidos por los clientes o usuarios finales. Para ello, es muy importante el idioma en el que estén las palabras y textos que aparezcan en tales modelos, por esto es que permite que todo modelo se cree en español o en cualquier otro lenguaje.

"UML ayuda al usuario a entender la realidad de la tecnología y la posibilidad de que reflexione antes de invertir y gastar grandes cantidades en proyectos que no estén seguros en su desarrollo, reduciendo el coste y el tiempo empleado en la construcción de las piezas que constituirán el modelo. Sin embargo, desde el punto de vista puramente tecnológico, UML tiene una gran cantidad de propiedades que han sido las que realmente han contribuido a hacer de UML un estándar de la industria.

Algunas de las propiedades de UML como lenguaje de modelado estándar son:

- Es un lenguaje distribuido y adecuado a las necesidades actuales y futuras de conectividad.
- Ampliamente utilizado por la industria desde su adopción por OMG.
- Reemplaza a decenas de notaciones empleadas por otros lenguajes.
- Modela estructuras complejas.
- Las estructuras más importantes que soportan tienen su fundamento en las tecnologías orientadas a objetos, tales como objetos, clase, componentes y nodos.
- Emplea operaciones abstractas como guía para variaciones futuras, añadiendo variables si es necesario.

- Realiza un comportamiento del sistema en casos de uso, diagramas de secuencia y de colaboraciones, que sirven para evaluar el estado de las máquinas.” [3]

UML ha generado y sigue generando un enorme entusiasmo comparable al nacimiento del desarrollo orientado a objetos a principios de los 90 y posteriormente el desarrollo de componentes en la segunda mitad de la década. UML se ha convertido ya en una de las mejores herramientas para el diseño y desarrollo de software fiable, eficiente y de calidad.

1.4.2 Lenguajes de Programación Web.

Los lenguajes de programación Web se dividen fundamentalmente en dos grupos, los del lado del cliente y los del lado del Servidor.

Entre los lenguajes del lado del servidor podemos encontrar, PERL, ASP, PHP, Java, que son los más usados en la actualidad. Son los que se desarrollan dentro del servidor, y los encargados del acceso a bases de datos, del tratamiento de la información, del desarrollo de aptitudes de gráfico Web en los programadores, etc.

Del lado del cliente se encuentran principalmente JavaScript (JScript) y el Visual Basic Script (VBScript), este último generalmente usado a la hora de programar en ASP. Estos lenguajes son los encargados de aportar dinamismo a la aplicación en los navegadores.

1. Perl (Practical Extracting and Reporting Language).

Perl es un lenguaje de propósito general originalmente desarrollado para la manipulación de texto y que hoy en día es utilizado para un amplio rango de tareas incluyendo administración de sistemas, desarrollo web, programación en red, entre otros.

Perl es un lenguaje interpretado en el servidor, o sea, que el código de los scripts en Perl no se compila, sino que cada vez que se quiere ejecutar se lee el código, interpretando lo que hay escrito en él, y el intérprete de Perl compila el programa a código máquina de forma transparente al usuario. Además es extensible a otros lenguajes, ya que desde Perl podremos hacer llamadas a subprogramas escritos en otras herramientas y a su vez, ejecutar desde ellas el código Perl. Suele ser muy práctico para extraer información de archivos de texto y generar informes a partir del

contendido de los ficheros. Es de uso gratuito y antes estaba muy asociado a la plataforma Unix, pero en la actualidad está disponible en otros sistemas operativos como Windows.

Una de las diferencias fundamentales de Perl con otros lenguajes es que no limita el tamaño de los datos con los que trabaja sino que el límite lo pone la memoria que en ese momento se encuentre disponible.

2. ASP (Active Server Pages).

ASP es un lenguaje desarrollado por Microsoft para la creación de páginas dinámicas y es interpretado en el servidor. Los tipos de servidores que emplea este lenguaje son aquellos que funcionan con sistema operativo de la familia de Windows NT.

Se escribe en la misma página Web, utilizando el lenguaje Visual Basic Script o Jscript (Javascript de Microsoft).

Actualmente se ha presentado la segunda versión de ASP, el ASP.NET, este tiene algunas diferencias en cuanto a sintaxis con el ASP, de modo que se ha de tratar de distinta manera. Para implementar ASP.NET es necesario montar en el Servidor la Plataforma .NET. Este además, permite el uso de una gran variedad de lenguajes de programación y, por tanto, se puede escoger el mejor lenguaje según las necesidades de la aplicación, o dividir su solución en varios lenguajes.

A pesar de su amplia aplicación en la actualidad, no es conveniente usar ASP para el desarrollo de la Intranet pues solo se puede implementar en los Servidores Web de Microsoft, con todas las implicaciones de alto costo, poca flexibilidad y escasa seguridad que estos conllevan, aspecto por el cual su desarrollo se ha frenado ante otros lenguajes.

3. JavaScript.

JavaScript es un lenguaje de programación utilizado para crear pequeños programas encargados de realizar determinadas acciones dentro del ámbito de una página Web.

Se trata de un lenguaje de programación del lado del cliente porque es el navegador del cliente el encargado de interpretar las instrucciones JavaScript y ejecutarlas, de modo que el mayor recurso con que cuenta este lenguaje es el propio navegador.

Permite la programación de pequeños scripts, pero también de programas más grandes, orientados a objetos, con funciones y estructuras de datos complejas. Por otra parte pone a disposición del programador todos los elementos que forman la página Web, para que éste pueda acceder a ellos y modificarlos dinámicamente.

”Como JavaScript es muy sencillo de aprender, se puede empezar a trabajar con él desde el principio. Es ideal para agregar ciertas funciones rápidas a una página Web. Una vez que se conocen las bases del lenguaje, no hay que esforzarse mucho para crear grandes aplicaciones. JavaScript también es un lenguaje muy potente de alto nivel. Es cierto que no se puede hacer nada directamente al nivel de la máquina, pero es capaz de trabajar con muchas propiedades de los exploradores Web, páginas Web y, a veces, con el propio sistema donde se ejecuta el explorador. No necesita una fase de compilación como Java o C y el explorador no ha de cargar ninguna máquina virtual para ejecutar el código. Solo hay que crear el código y cargarlo. También se trabaja con propiedades como las funciones del constructor o la estratificación en jerarquías, por lo que hay más opciones para utilizar el código.” [4]

Este lenguaje se usará en la Intranet fundamentalmente para la creación de efectos especiales en las páginas y la definición de interactividades con el usuario.

4. PHP (Hypertext Preprocessor).

Después de haber hecho un análisis de varios de los lenguajes mas utilizados actualmente para el desarrollo de aplicaciones Web se decide por el lenguaje de programación de páginas dinámicas del lado del servidor, multiplataforma y totalmente gratuito PHP, perteneciente a la familia de software libre, característica muy conveniente hoy en día, pues el software libre es aquel que, una vez obtenido, puede ser usado, copiado, estudiado, modificado y redistribuido libremente. Además el software libre suele estar disponible gratuitamente en Intranet, o a precio del coste de la distribución a través de otros medios; sin embargo no es obligatorio que sea así y, aunque conserve su carácter de libre, puede ser vendido comercialmente.

PHP es un lenguaje multiplataforma, pues puede ejecutarse en entornos Unix, Linux y Windows, y es independiente del navegador que utilices, pues el código PHP se ejecuta en el servidor y al navegador solo devuelve texto.

Contamos además con PHP para la programación del sistema porque es usado generalmente para la creación de contenido dinámico en aplicaciones Web y para la

creación de programas incluyendo aplicaciones con interfaz gráfica. Además, es un soporte sólido para la Programación Orientada a Objetos. “Permite una serie de funcionalidades:

Gratuito: Al tratarse de software libre, como se mencionó anteriormente, puede descargarse y utilizarse en cualquier aplicación, personal o profesional, de manera completamente libre.

Gran popularidad: Existe una gran comunidad de desarrolladores y programadores que continuamente implementan mejoras en su código, y que en muchos casos estarán dispuestos a ayudar cuando exista algún problema. Existen millones de páginas Web desarrolladas con PHP.

Enorme eficiencia: Con escaso mantenimiento y un servidor gratuito (en nuestro caso, Apache), puede soportar sin problema millones de visitas diarias. Sencilla integración con múltiples bases de datos lo cual es esencial para una página Web verdaderamente dinámica. MySQL es la base de datos que mejor trabaja con PHP, pero puede conectarse también a PostgreSQL, Oracle, entre otros.

Versatilidad: PHP puede usarse con la mayoría de los sistemas operativos, ya sea basados en UNIX (Linux, Solaris, FreeBSD), como con Windows, es decir, como se explicó anteriormente multiplataforma.

Gran número de funciones predefinidas: A diferencia de otros lenguajes de programación, PHP fue diseñado especialmente para el desarrollo de páginas Web dinámicas. Por ello, está dotado de un gran número de funciones que nos simplificarán enormemente tareas habituales como descargar documentos, enviar correos, trabajar con cookies y secciones, etc.” [5]

PHP soporta código XML, es compatible con MySQL gracias a los iteradores de datos que contiene que le permiten conectarse desde la aplicación Web a la base de datos, hacer consultas, agregar datos e incluso ingresar al área de administración de MySQL y modificarla. Además es compatible con el Servidor Web Apache que ofrece mucha seguridad para los sistemas Web y por ello es uno de los más usados en la actualidad.

1.5 SGBD (Sistemas de Gestión de Bases de Datos).

Los sistemas de gestión de base de datos son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. Se compone de un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta. En los textos que tratan este tema, se mencionan los términos SGBD y DBMS, siendo ambos equivalentes, y acrónimos, respectivamente, de Sistema Gestor de Bases de Datos. El propósito general de los sistemas de gestión de base de datos es manejar de manera clara, sencilla y ordenada un conjunto de información.

Las principales funciones de un SGBD están relacionadas con la creación y mantenimiento de la base de datos, el control de accesos, la manipulación de datos de acuerdo con las necesidades del usuario, el cumplimiento de las normas de tratamiento de datos, evitar redundancias e inconsistencias y mantener la integridad.

Entre los SGBD comúnmente utilizados tenemos Oracle, MySQL, Microsoft SQL Server, PostgreSQL, entre otros. Todos estos presentan un enfoque relacional con un buen basamento matemático centrado en el Álgebra Relacional.

1.5.1 Oracle.

Oracle es un sistema gestor de base de datos fabricado por Oracle Corporation. Se considera como uno de los sistemas de bases de datos más completos, destacándose por su soporte de transacciones, entidades, escalabilidad y multiplataforma.

“Oracle enfatiza los tres principales objetivos de su nueva tecnología objeto relacional de bases de datos, que son los siguientes:

- Proporcionar la posibilidad de que el usuario modele objetos de negocio en la base de datos permitiendo la definición y el uso de tipos definidos por el usuario.
- Proporcionar infraestructura para el acceso a base de datos.
- Permite mantener el modelo de datos estrechamente cercano al que se usa en las aplicaciones del mundo real.” [6]

Los tipos de objetos de Oracle son tipos de datos definidos por el usuario que permiten modelar entidades complejas del mundo real en una estructura que trata cada entidad como una unidad atómica simple en la base de datos.

“Las aplicaciones que utilizan objetos de Oracle son fáciles de entender y mantener ya que los tipos objeto de Oracle soportan totalmente las características de abstracción y encapsulación del comportamiento de los objetos basados en el paradigma orientado a objetos.” [6]

El principal motivo por el cual se ha decidido no utilizarlo como sistema gestor de base de datos de la Intranet es su enorme precio. También otro aspecto criticado por algunos especialistas es la seguridad de la plataforma, y las políticas de suministro de parches de seguridad, que incrementan el nivel de exposición de los usuarios.

Aunque su dominio en el mercado de servidores empresariales ha sido casi total, recientemente sufre la competencia del Microsoft SQL Server y de la oferta de otros SGBD con licencia libre como PostgreSQL y MySQL.

1.5.2 Microsoft SQL Server.

Microsoft SQL Server es un sistema de gestión de base de datos basado en el lenguaje SQL, capaz de poner a disposición de muchos usuarios grandes cantidades de datos de manera simultánea.

Posee ventajas que se pueden describir a continuación como:

- Soporte de transacciones.
- Gran estabilidad.
- Gran seguridad.
- Escalabilidad.
- Soporta procedimientos almacenados.
- Permite trabajar en modo cliente- servidor donde la información y datos se alojan en el servidor y las terminales o clientes de la red sólo acceden a la información.
- Posibilita administrar información de otros servidores de datos.

Las utilidades de administración de este gestor son envidiables para muchos de los gestores comerciales existentes, debido a su gran facilidad de configuración e instalación. Es utilizada por una gran cantidad de usuarios lo que la hace muy popular y de fácil acceso.

No es multiplataforma, pues sólo está disponible en Sistemas Operativos de Microsoft, constituyendo la principal desventajas de este gestor de base de datos.

1.5.3 PostgreSQL.

PostgreSQL está considerado como el sistema de gestión de bases de datos de código abierto más avanzado del mundo. PostgreSQL proporciona un gran número de funcionalidades que normalmente sólo se encontraban en sistemas de bases de datos comerciales tales como Oracle.

“Algunas de sus principales características son:

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 multiusuario, optimización de consultas, herencia, y arreglos.

Altamente Extensible: PostgreSQL soporta operadores, funcionales 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).

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

API Flexible: La flexibilidad del API de PostgreSQL ha permitido a los vendedores proporcionar soporte al desarrollo fácil para el RDBMS PostgreSQL. Estas interfaces incluyen Python, Perl, PHP, ODBC, Java/JDBC, TCL, C/C++, etc” [7]

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. Otra ventaja de PostgreSQL es su habilidad para usar Perl, Python, o TCL como lenguaje procedural embebido.

MVCC: MVCC, o Control de Concurrencia Multi-Versión (Multi-Versión Concurrency Control), es la tecnología que PostgreSQL usa para evitar bloqueos innecesarios. Si alguna vez se usa algún DBMS con capacidades SQL, tal como MySQL o Access, probablemente se note que hay ocasiones en las que una lectura tiene que esperar para acceder a información de la base de datos. La espera está provocada por usuarios que están escribiendo en la base de datos. Mediante el uso de MVCC, PostgreSQL evita este tipo de problemas. MVCC está considerado mejor que el bloqueo a nivel de fila porque un lector nunca es bloqueado por un escritor. En su lugar, PostgreSQL mantiene una ruta a todas las transacciones realizadas por los usuarios de la base de datos. PostgreSQL es capaz entonces de manejar los registros sin necesidad de que los usuarios tengan que esperar a que los registros estén disponibles.

Cliente/Servidor: PostgreSQL usa una arquitectura proceso-por-usuario cliente/servidor. Hay un proceso maestro que se ramifica para proporcionar conexiones adicionales para cada cliente que intente conectar a PostgreSQL.

Write Ahead Logging (WAL): La característica de PostgreSQL conocida como Write Ahead Logging incrementa la dependencia de la base de datos al registro de cambios antes de que estos sean escritos en la base de datos. Esto garantiza que en el hipotético caso de que la base de datos se caiga, existirá un registro de las transacciones a partir del cual podremos restaurar la base de datos. Esto puede ser enormemente beneficioso en el caso de una caída, ya que cualesquiera cambios que no fueron escritos en la base de datos pueden ser recuperados usando el dato que fue previamente registrado. Una vez el sistema ha quedado restaurado, un usuario puede continuar trabajando desde el punto en que lo dejó cuando cayó la base de datos.

A pesar de estar Postgresql considerado como el mejor sistema de gestión de base de datos de código abierto, la Intranet que se desarrollará es sencilla y no se necesita de un sistema tan complejo para gestionar su base de datos, además los servidores donde se alojará la misma poseen pocas prestaciones y con un sistemas menos complejo se garantiza el buen funcionamiento de la gestión de los datos.

1.5.4 MYSQL.

MySQL es un sistema de gestión de bases de datos relacional, que es capaz de almacenar una enorme cantidad de datos de gran variedad. Utiliza el lenguaje de consulta estructurado SQL que es el lenguaje utilizado por todas las bases de datos

relacionales, que permite crear bases de datos, así como agregar, manipular y recuperar datos en función de criterios específicos.

MySQL fue seleccionado como sistema de gestión de base de datos para el desarrollo de la Intranet, por los buenos resultados que en la práctica se han obtenido complementado con PHP y el servidor Web Apache, además, compite con sistemas como Oracle y SQL Server. Incluye todos los elementos necesarios para instalar el programa, preparar diferentes niveles de acceso de usuario, permite administrar el sistema y protegerlo, así como, hacer volcados de datos. Puede desarrollar sus propias aplicaciones de base de datos en la mayor parte de los lenguajes de programación utilizados en la actualidad y ejecutarlos en casi todos los sistemas operativos.

Por otra parte, permite contar con la disponibilidad de otras plataformas y sistemas, la realización de diferentes opciones de almacenamiento según si se desea velocidad en las operaciones o el mayor número de operaciones disponibles, además, el poder instalar y posteriormente desinstalar tantas versiones de MySQL como se desee comprobar, así como, beneficiarnos con su poca capacidad de almacenamiento.

MySQL permite a los usuarios cambiar opciones de arranque sin tener que reiniciar el servidor, la replicación a prueba de fallos y backup en línea con poca penalización en el rendimiento.

1.6 Servidor Web Apache.

Después de haber seleccionado como lenguaje de programación PHP y como gestor de base de datos a MySQL, la mejor elección para el servidor Web es Apache, pues este trío constituye la combinación perfecta para el desarrollo de aplicaciones Web.

Apache es un programa de servidor HTTP Web de código abierto (open source). Fue desarrollado en 1995 y actualmente es uno de los servidores Web más utilizados en la red. Usualmente corre en UNIX, Linux y Windows. Es un poderoso paquete de servidor Web con muchos módulos que se le pueden agregar y que se consiguen gratuitamente en Internet.

“Presenta entre otras características mensajes de error altamente configurables, bases de datos de autenticación y negociado de contenido, pero tiene como deficiencia, la falta de una interfaz gráfica que ayude a su configuración.

El servidor de HTTP Apache es un proyecto de software colaborativo, dirigido a poner en práctica una aplicación robusta. El proyecto es manejado en común por un grupo de voluntarios situados alrededor de todo el mundo, usando Internet y la Web para comunicarse, planear, desarrollar el servidor y su documentación relacionada. Este proyecto es parte de la Fundación del Software Libre y centenares de usuarios han contribuido a él con ideas, código, y documentación." [8]

Contiene la infraestructura necesaria para servir distintos protocolos, es más rápido y más estable en sistemas que no son tipo Unix, como Windows, además, tiene implementada su propia API nativa, evitando que las capas de emulación POSIX provoquen problemas y bajos rendimientos.

Por otra parte los módulos de Apache pueden escribirse para que se comporten como filtros que actúan sobre el flujo de contenidos tal y como salen del servidor o tal y como son recibidos por el servidor. Los mensajes de error que se envían a los navegadores están disponibles en diferentes idiomas e incluye la librería de expresiones regulares compatibles con el lenguaje Perl.

1.7 Metodologías.

Una metodología es un proceso que define un conjunto ordenado de pasos a seguir para cumplir un objetivo. En la ingeniería de software, la metodología define quién debe hacer, qué, cuándo y cómo debe hacerlo.

Las metodologías guían el proceso de desarrollo y la experiencia ha demostrado que la clave del éxito de un proyecto de software es la elección correcta de esta, pues puede conducir al programador a desarrollar un buen sistema de software.

La elección de la metodología adecuada es más importante que utilizar las mejores y más potentes herramientas. La idea no es tratar de ver cual es mejor o peor, sino de cuando usar una y cuando la otra, pues esto va de acuerdo al tipo de proyecto, a los recursos con los que se cuentan (tiempo, dinero, etc) y a la facilidad de interacción con el usuario real.

Las metodologías que existen actualmente se adecuan al desarrollo de la mayoría de las aplicaciones, puesto que han surgido de la experiencia en la producción acumulada por varios años, dando lugar a la calidad del software requerida.

1.7.1 Metodología Ágil XP (Extreme Programming).

La metodología XP permite establecer iteraciones muy cortas, apropiada para un entorno caracterizado por requerimientos cambiantes. Su objetivo principal es tener una nueva versión a cada instante, mostrarlo al cliente, ver lo que opina y seguir programando, tener una comunicación fluida con el cliente y el usuario final, es decir, define una manera de reunir a clientes y programadores en un equipo firmemente integrado con condiciones de trabajo que promueven la comunicación y solución de un problema. Se ha clasificado como una metodología ágil, ya que plantea aumentar constantemente la velocidad del proyecto.

"El desarrollo bajo la metodología XP tiene características que lo distinguen claramente de otras metodologías:

- Los diseñadores y programadores se comunican efectivamente con el cliente y entre ellos mismos.
- Los diseños del software se mantienen sencillos y libres de complejidad o pretensiones excesivas.
- Se obtiene retroalimentación de usuarios y clientes desde el primer día gracias a las baterías de pruebas.
- El software es liberado en entregas frecuentes tan pronto como sea posible.
- Los cambios se implementan rápidamente tal y como fueron sugeridos.
- Las metas en características, tiempos y costos son reajustadas permanentemente en función del avance real obtenido." [7]

Esta metodología funciona mejor para pequeños equipos, a diferencia de RUP, que es muy optima para un equipo grande de desarrolladores, esto sin lugar a dudas lo pone en desventaja, por otra parte también, reducen la documentación asociada a cada proyecto (esto puede ser riesgoso con proyectos de complejidad alta). No cuenta prácticamente con una documentación para su constancia, lo que implica que los desarrolladores se acostumbren a esta mala práctica. Estas son las principales razones por las que no se seleccionó esta metodología para el desarrollo de la aplicación.

1.7.2 Metodologías de Desarrollo Rational Unified Process (RUP).

La metodología ideal para el desarrollo de la Intranet es la RUP. “Metodología de desarrollo de software orientada a objetos que proporciona un método disciplinado para asignar las tareas y responsabilidades dentro del equipo de desarrollo. Su objetivo es asegurar la producción de software de alta calidad que resuelva las necesidades del usuario dentro de un cronograma predecible y al menor costo posible.” [9] Está basado en componentes y utiliza el lenguaje UML para visualizar, especificar y documentar cada una de las partes que comprende el desarrollo de software.

“RUP es un proceso de desarrollo de software. Un proceso de desarrollo de software es el conjunto de actividades necesarias para transformar los requisitos de un usuario en un sistema de software. Sin embargo, el proceso unificado es más que un proceso de trabajo, es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas software, para diferentes áreas de aplicación, diferentes tipos de organizaciones y diferentes niveles de aptitud”. [9]

Es muy conveniente esta metodología porque durante su ciclo de vida se desarrolla toda una metodología iterativa incremental que va eliminando los errores cometidos en las iteraciones previas, logrando que al final del proceso se obtenga como resultado un producto de calidad. Está dirigido por casos de uso que son los que guían todo el proceso de desarrollo, permitiendo obtener lo que los usuarios futuros necesitan y desean. Centrado en la arquitectura posibilitando describir los elementos más importantes para su construcción, los cimientos del sistema que son necesarios como base para comprenderlo, desarrollarlo y producirlo económicamente, mostrando al final, la visión del sistema completo. De esta manera se obtiene los resultados de un proceso de ingeniería de software entendible tanto por el equipo de desarrollo como por el cliente. Por otra parte, define los roles a jugar por cada miembro del equipo de desarrollo en cada una de las etapas por las que transcurre el sistema, facilitando la comunicación entre los diferentes miembros del equipo de desarrollo.

Esta metodología es ideal para la gestión de los requisitos, desarrollo visual del software (con UML), la verificación continua de la calidad del software, la gestión de los cambios y sobre todo, para proyectos grandes, a largo plazo y con un equipo de desarrollo numeroso.

1.8 Herramienta para la Modelación Visual del Sistema.

Las herramientas CASE (Computer Aided Software Engineering), utilizan Unified Modeling Language (UML). Pueden ser generalmente aplicadas a cualquier sistema o colección de herramientas que permita automatizar el proceso de diseño y desarrollo de software.

Estas herramientas están destinadas a aumentar la productividad en el desarrollo de software reduciendo el coste en términos de tiempo y de dinero. Pueden ayudar en todos los aspectos del ciclo de vida de desarrollo del software en tareas como el proceso de realizar un diseño del proyecto, calculo de costes, implementación de parte del código automáticamente con el diseño dado, compilación automática y documentación o detección de errores entre otras.

"Cuando se hace la planificación de la base de datos, la primera etapa del ciclo de vida de las aplicaciones de bases de datos, también se puede escoger una herramienta CASE que permita llevar a cabo el resto de tareas del modo más eficiente y efectivo posible. Una herramienta CASE suele incluir:

- Un diccionario de datos para almacenar información sobre los datos de la aplicación de bases de datos.
- Herramientas de diseño para dar apoyo al análisis de datos.
- Herramientas que permitan desarrollar el modelo de datos corporativo, así como los esquemas conceptual y lógico.
- Herramientas para desarrollar los prototipos de las aplicaciones.
- El uso de las herramientas CASE puede mejorar la productividad en el desarrollo de una aplicación de bases de datos." [10]

1.8.1 Rational Rose.

Rational Rose es una herramienta CASE de modelación visual desarrollada por Rational Corporación basada en UML, que permite crear los diagramas que se van generando durante el proceso de ingeniería en el desarrollo de un sistema informático. Brinda facilidades para la generación de la documentación del software que se está desarrollando y posee un gran número de estereotipos predefinidos que agilizan el proceso de modelación.

Acelera el desarrollo a través del modelado visual y las funciones de generación de código e ingeniería inversa, por otra parte, encuentra y elimina errores de ejecución, pérdidas de memoria y cuestiones de rendimiento. Posibilita capacidades avanzadas de modelado visual para bases de datos.

Dicha herramienta es capaz de generar el código fuente de las clases definidas en el flujo de trabajo de diseño, pero tiene la limitación de que aún hay varios lenguajes de programación que no soporta o que sólo lo hace a medias. Por otra parte, una vez terminado el diagrama de clases persistentes a partir del cual se genera la base de datos del sistema, no existe la posibilidad de que el mismo exporte ese modelo hacia algún sistema gestor de bases de datos. Es la herramienta usada generalmente para el desarrollo de software propietario, siendo esta una característica significativa que dio lugar a que no se seleccionara para el modelado de la Intranet.

1.8.2 Visual Paradigm.

El Visual Paradigm es una poderosa herramienta CASE de modelación visual. Utiliza UML para el modelado, permitiendo crear tipos diferentes de diagramas en un ambiente totalmente visual. Es muy sencillo de usar, fácil de instalar y actualizar. Genera código para varios lenguajes. Es una herramienta que puede ser utilizada en la creación de software libre, es nombrada por muchas bibliografías como la herramienta CASE por excelencia del software libre, siendo esta una de las características que dio lugar a que se seleccionara para el desarrollo de la Intranet, sin dejara de mencionar que el “Visual Paradigm, además posee:

- Un entorno de creación de diagramas para UML.
- Diseño centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad.
- Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- Capacidades de ingeniería directa e inversa.
- Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
- Disponibilidad de múltiples versiones, para cada necesidad.
- Disponibilidad de integrarse en los principales IDEs.

- Disponibilidad en múltiples plataformas.” [11]

Por otra parte, posibilita la representación gráfica de los diagramas permitiendo ver el sistema desde diferentes perspectivas, como el de componentes, despliegue, secuencia, casos de uso, clase, actividad, entre otros. Además, se centra en cómo los componentes del sistema interactúan entre ellos, sin entrar en detalles excesivos, también, permite ver las relaciones entre los componentes del diseño y mejora la comunicación entre los miembros del equipo usando un lenguaje gráfico.

Tiene disponible distintas versiones: Enterprise, Professional, Standard, Modeler, Personal y Community. Facilita licencias especiales para fines académicos.

Conclusiones

- Se realizó un estudio de aplicaciones Web tanto nacionales como internacionales que presentan similitud con el sistema a desarrollar.
- Se presentó los aportes prácticos esperados de la Intranet que se pretende diseñar.
- Se describieron las características de herramientas y metodologías para la elección de las que se usarán en el desarrollo de la aplicación, así como conceptos asociados al dominio del problema.
- Se seleccionó RUP como metodología para el modelado del sistema con UML como lenguaje de representación visual.
- Se seleccionó PHP, MYSQL y Apache como herramientas básicas para la creación de la Intranet, debido a que la unión de estas tecnologías ha demostrado ofrecer eficientes resultados.

Capítulo II

Características del Sistema

En este capitulo se describirán los procesos que serán objetos de automatización y se definirán los requerimientos funcionales y no funcionales con que el sistema debe cumplir para satisfacer las necesidades del cliente.

Se definirán además los casos de uso y actores del sistema y se representarán gráficamente las relaciones entre ambos en el diagrama de casos de uso del sistema. Finalmente se describirán los casos de uso, teniendo en cuenta las acciones del actor sobre el sistema y las respuestas que el sistema ejecuta ante cada acción.

2.1 Situación Problémica.

Con la ayuda de la Intranet, las Residencias de Protocolo podrán implementar un flujo de contenido que permita a los huéspedes que la visiten informarse sobre la historia de las residencias, los servicios que se les brinda, entre otros aspectos de interés, además se les da la posibilidad de realizar en línea la solicitud del servicio de lavandería, responder encuestas y ofrecer opiniones que reflejen su grado de satisfacción con la atención que han recibido.

Debido a la relevancia de los huéspedes que se reciben en las Residencias, el personal seleccionado para atenderlos está capacitado para ofrecerles toda la información que él debe conocer, además la mayoría de los huéspedes cuentan con un programa de las actividades que realizará durante su visita a Cuba y muchas veces incluye detalles generales de su hospedaje en las residencias, pero una vez que se pueda contar con la Intranet, tendrá esta, entre sus funciones especificas, centralizar y organizar toda la información que además estará disponible en los idiomas más hablados mundialmente.

Por otra parte en la actualidad, los huéspedes solicitan el servicio de lavado y planchado al ama de llaves y ella es la encargada de informar a la lavandería de la solicitud, una vez que funcione la Intranet el huésped podrá realizar la solicitud directamente a la lavandería de manera online.

Comúnmente los huéspedes al finalizar su estadía en las Residencias utilizan el Libro de Opiniones para dejar plasmadas sus impresiones sobre su visita, también podrán

hacerlo a través de la Intranet así como realizar alguna queja y contestar las encuestas que estarán orientadas siempre a incrementar el confort de las Residencias.

2.2 Objeto de Automatización.

Se automatizarán los procesos de difusión de la información relacionada con las Residencias de Protocolo, la solicitud del servicio de lavandería y el llenado de encuestas y opiniones.

2.3 Información que se Maneja.

La información que formará parte de los contenidos de la Intranet serán: la historia de las Residencias de Protocolo, información de los servicios que se ofertan a los huéspedes en las residencias, datos sobre las residencias, el sitial histórico, la figura de Celia Sánchez Manduley y las obras de arte que decoran las residencias.

2.4 Propuesta de Sistema.

En aras de satisfacer los requerimientos planteados se propone un sistema que conste de dos módulos: uno que será el punto de acceso a toda la información y funcionalidades de la Intranet y un módulo administrativo donde se realizarán las actualizaciones de los contenidos.

La Intranet estará disponible solamente para los huéspedes y los directivos de las Residencias de Protocolo que podrán navegar e interactuar con ella sin ninguna restricción, mientras que la administración de los contenidos se realizará solamente por el administrador o alguna persona autorizada por este.

2.5 Modelo de Negocio.

Después de entender los procesos y la estructura de la organización de las Residencias de Protocolo, de haber realizado un estudio del negocio que se desarrolla se puede apreciar que no existen procesos que estén claramente definidos, mientras que existen otros con un bajo nivel de estructuración y soluciones dispersas. Estos procesos se presentarán capturando los tipos más importantes de objetos que existen o los eventos que suceden en el entorno donde estará el sistema, es decir, se utilizará el modelo de dominio que permitirá de manera visual mostrar al usuario los principales conceptos que se manejan en el dominio del sistema en desarrollo, ayudando a los usuarios, clientes, desarrolladores e interesados, a utilizar un vocabulario común para poder entender el contexto en que se emplaza el sistema.

Para el desarrollo del Modelo de Dominio se comienza identificando todos los conceptos que se utilizarán en el diagrama, mediante un glosario de términos:

Se considerará **Huésped** a la persona que habita la residencia durante un espacio de tiempo.

Se considerará **Residencia** a la casa que aloja al huésped.

Se considerará **Obra de Arte** a una obra de arte ubicada en una Residencia.

Se considerará **Visita** a la presencia de un huésped en un sitio relacionado con la organización.

Se considerará **Servicio de Gastronomía** a las ofertas de alimentos, bebidas, vinos y cocteles para un huésped.

Se considerará **Servicio de Alojamiento** a la asignación de una residencia a un huésped.

Se considerará **Servicio de Transporte** a la asignación de un auto a un huésped.

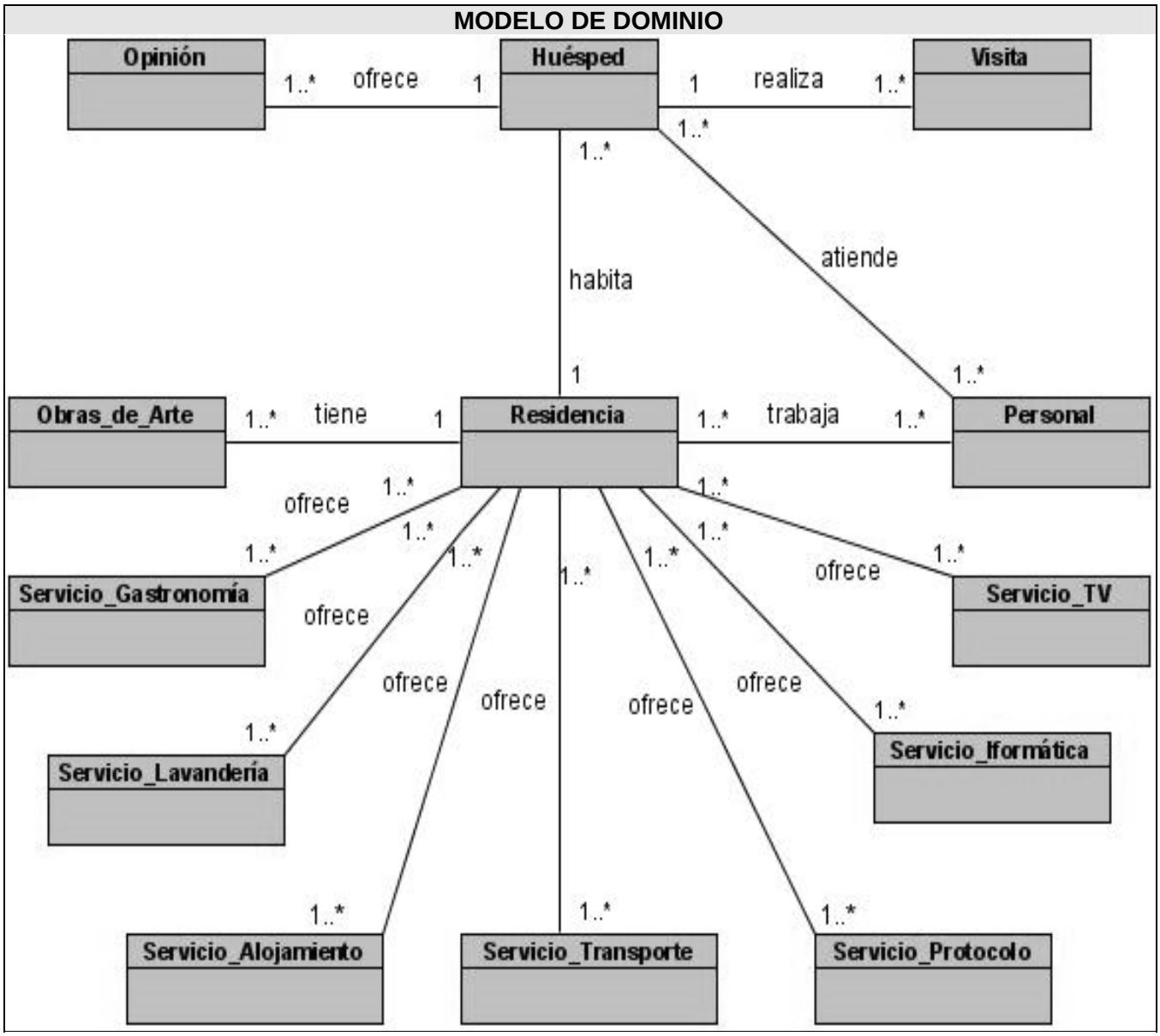
Se considerará **Servicio de Protocolo** a la oferta de salón diseñado para desarrollar actividades políticas y culturales de pequeño y gran formato.

Se considerará **Servicio de Informática** a los servicios informáticos que se le ofrece al huésped.

Se considerará **Servicio de Televisión** a la oferta de canales televisivos para el huésped.

Se considerará **Personal** a la persona que trabaja en las Residencias de Protocolo.

Se considerará **Opinión** la opinión que ofrece el huésped durante su estancia en la residencia.


MODELO DE DOMINIO
Opinión
Huésped
Visita
1..* ofrece 1
1 realiza 1..*
1..*
1..*
habita
atiende
1
1..*
Obras_de_Arte
1..* tiene 1
Residencia
1..* trabaja 1..*
Personal
ofrece
1..*
1..*
1..*
1..*
1..*
1..*
1..*
1..*
Servicio_Gastronomía
Servicio_TV
ofrece
ofrece
ofrece
ofrece
ofrece
ofrece
1..*
1..*
Servicio_Lavandería
Servicio_Iformática
1..*
1..*
1..*
Servicio_Alojamiento
Servicio_Transporte
Servicio_Protocolo

2.6 Especificación de los Requisitos del Software.

2.6.1 Requerimientos Funcionales.

Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir. Se mantienen invariables sin importar con que propiedades o cualidades se relacionen.

R1 Configurar Idioma:
El sistema debe permitir al usuario seleccionar en que idioma desea que se muestre la información.

R2 Contactar:
El sistema debe ofrecer al usuario un vínculo a una dirección de correo electrónico para establecer un contacto.

R3 Visualizar Información

R3.1 Visualizar información general de las residencias:
El sistema debe mostrar la información referente a las residencias:
- Interiores de la residencia.
- Personal que labora en la residencia.
- Información general de los servicios que se ofertan.
- Mapa de la residencia.
- Zonas donde se encuentran agrupadas las residencias.

R3.2 Visualizar información de las Zonas Residenciales:
El sistema debe mostrar el listado de residencias organizadas por zonas residenciales y mostrar de cada una la información:
- Historia de la residencia.
- Foto de la residencia.

R.3.3 Visualizar información general de los Servicios:
El sistema debe mostrar la información referente a los servicios de:
- Alojamiento.
- Gastronomía.
- Transporte.
- Protocolo.
- Informática.
- Lavandería.
- Televisión por cable.

R3.4 Visualizar la información del Servicio de Gastronomía:
El sistema debe mostrar la información referente al servicio de gastronomía:
- Menú del almuerzo y la cena organizado por el día de la semana.
- Vinos organizados por categorías.
- Bebidas organizadas por categorías.
- Cocteles agrupados en cocteles nacionales y cocteles internacionales.

R3.5 Visualizar la información de Sitios Web cubanos:
El sistema debe mostrar un listado de vínculos (cada uno con su descripción) a sitios cubanos organizados por las temáticas:
- Salud.
- Prensa nacional.
- Turismo.

- Cultura.
- Gobierno.
- Educación.
- Ciencia.
- Eventos.

R3.6 Visualizar la información del Sitial Histórico:
El sistema debe mostrar la información referente al Sitial Histórico:
- Especificaciones del lugar.
- Biografía de Celia Sánchez Manduley.

R3.7 Visualizar las imágenes de la Galería:
El sistema debe mostrar el conjunto de fotos organizadas por las secciones:
- Residencia.
- Platos ofertados por la casa.
- Bebidas ofertadas por la casa.
- Cocteles ofertados por la casa.
- Vinos ofertados por la casa.
- Vida de Celia Sánchez Manduley.

R3.8 Visualizar la información de las Obras de Arte:
El sistema mostrará solo la información de las obras de arte que se encuentren en la residencia desde la cual está accediendo el usuario, la información que mostrará será:
- Imagen de la obra.
- Título de la obra.
- Fecha de la obra.
- Origen de la obra.
- Época de la obra.
- Autores de la obra.
- Descripción de la obra.
- Manifestación de la obra.
- Denominación de la obra.
- Técnica de la obra.
- Medidas de la obra.
- Materiales de la obra.
- Soporte de la obra.
- Color de la obra.
- Estado de la obra.
- Valor de la obra.
- Grado de valor de la obra.
- Tasación de la obra.
- Ubicación actual de la obra.
- Catalogador de la obra.
- Número de inventario de la obra.
- Temática de la obra.
- Lugar de la firma de la obra.
- Escuela del autor de la obra.
- Casa editora de la obra.
- Forma de la obra.

R3.9 Visualizar Mapa de la Intranet:
El sistema debe visualizar al usuario un mapa donde se encuentre la estructura de la navegabilidad de la Intranet.

R4 Solicitar el Servicio de Lavandería:
Esta funcionalidad estará disponible todos los días, antes de las 9 horas, luego el sistema debe deshabilitarla.

El sistema debe permitir que el usuario solicite el servicio de lavandería introduciendo los datos:
- Nombre.
- Servicio (lavado y/o planchado).

R5 Realizar Encuesta:
El sistema debe permitir al usuario realizar la encuesta introduciendo los datos:
- Nombre.
- Respuesta.

R6 Ofrecer Opinión:
El sistema debe permitir al usuario ofrecer una opinión introduciendo los datos:
- Nombre.
- Servicio del que desea opinar.
- Opinión.

R7 Restringir el acceso.
Se debe Controlar que cada usuario acceda a la información que le corresponda, para ello se debe definir niveles de acceso.

- Solo el administrador del sistema puede dar los permisos a otros usuarios para administrar la Intranet.

R8 Confeccionar Menú.

R8.1 Confeccionar Menú del Almuerzo:
El sistema debe permitir que el usuario seleccione el idioma en que desea confeccionar el menú de almuerzo.

El sistema debe permitir al usuario confeccionar el menú introduciendo la información:
- Fecha.
- Día de la semana.
- Entrante.
- Guarnición.
- Plato Fuerte.
- Postre.

R8.2 Confeccionar Menú de la Cena:
El sistema debe permitir que el usuario seleccione el idioma en que desea confeccionar el menú de la cena.

El sistema debe permitir al usuario confeccionar el menú introduciendo la información:
- Fecha.
- Día de la semana.
- Volante.
- Plato fuerte.
- Guarnición.
- Postre.

R8.3 Listar Menú:
El sistema debe permitir al usuario especificar en que idioma desea listar la información del menú.

El sistema debe permitir al usuario verificar la información referente al menú de almuerzo:
- Día de la semana.
- Entrante.
- Guarnición.
- Plato Fuerte.
- Postre.

El sistema debe permitir al usuario verificar la información referente al menú de la cena de cada día de la semana:
- Día de la semana.
- Volante.
- Plato fuerte.
- Guarnición.
- Postre**.**

R9 Gestionar Ingredientes.

R9.1 Listar Ingredientes:
El sistema debe permitir al usuario especificar en que idioma desea listar los ingredientes.

El sistema debe permitir al usuario verificar la información referente a los ingredientes:
- Nombre.
- Tipo de plato.

R9.2 Adicionar Ingredientes:
El sistema debe permitir al usuario especificar en que idioma desea adicionar los ingredientes.

El sistema debe permitir al usuario introducir la información referente a los ingredientes:
- Nombre.
- Tipo de plato.

R9.3 Modificar Ingredientes:
El sistema debe permitir al usuario especificar en que idioma desea modificar la información de los ingredientes.

El sistema debe permitir al usuario modificar la información referente a los ingredientes:
- Nombre.
- Tipo de plato.
- Idioma.

R9.4 Eliminar Ingredientes:
El sistema debe eliminar toda la información referente al ingrediente que el usuario desee suprimir.

R10 Gestionar Bebidas.

R10.1 Adicionar Bebidas:
El sistema debe permitir al usuario especificar en que idioma desea adicionar la bebida.

El sistema debe permitir al usuario introducir la información referente a la bebida:
- Imagen.
- Categoría.
- Nombre.
- Descripción.
- Descripción de la imagen.

R10.2 Listar Bebidas:
El sistema debe permitir al usuario especificar en que idioma desea listar las bebidas.

El sistema debe permitir al usuario verificar la información referente a las bebidas:
- Nombre.
- Descripción.

R10.3 Modificar Bebidas:
El sistema debe permitir al usuario especificar en que idioma desea modificar la información de las bebidas.

El sistema debe permitir al usuario modificar la información referente a la bebida:
- Imagen.
- Categoría.
- Nombre.
- Descripción.
- Descripción de la imagen.

R10.4 Eliminar Bebida:
El sistema debe eliminar toda la información referente a la bebida que el usuario desee suprimir.

R11 Gestionar Vinos.

R11.1 Adicionar Vinos:
El sistema debe permitir al usuario especificar en que idioma desea adicionar el vino.

El sistema debe permitir al usuario introducir la información referente al vino:
- Imagen.
- Categoría.
- Nivel.
- Nombre.
- Descripción de la imagen.

R11.2 Listar Vinos:
El sistema debe permitir al usuario especificar en que idioma desea listar los vinos.

El sistema debe permitir al usuario verificar la información referente a los vinos:
- Nombre.
- Nivel.

R11.3 Modificar Vinos:
El sistema debe permitir al usuario especificar en que idioma desea modificar la información del vino.

El sistema debe permitir al usuario modificar la información referente al vino:

- Imagen.
- Categoría.
- Nivel.
- Nombre.
- Descripción de la imagen.

R11.4 Eliminar Vino:
El sistema debe eliminar toda la información referente al vino que el usuario desee suprimir.

R12 Gestionar Cocteles.
R12.1 Adicionar Cocteles:
El sistema debe permitir al usuario especificar en que idioma desea adicionar el coctel.

El sistema debe permitir al usuario introducir la información referente al coctel:
- Nombre.
- Ingredientes.
- Modo de preparación.
- Imagen.
- Descripción de la imagen.

R12.2 Listar Cocteles:
El sistema debe permitir al usuario especificar en que idioma desea listar los cocteles.

El sistema debe permitir al usuario verificar la información referente a los cocteles:
- Nombre.
- Ingredientes.
- Modo de preparación.

R12.3 Modificar Cocteles:
El sistema debe permitir al usuario especificar en que idioma desea modificar la información del coctel.

El sistema debe permitir al usuario modificar la información referente al coctel:
- Nombre.
- Ingredientes.
- Modo de preparación.
- Imagen.
- Descripción de la imagen.

R12.4 Eliminar Coctel:
El sistema debe eliminar toda la información referente al coctel que el usuario desee suprimir.

R13 Gestionar Sitio Web.
13.1 Adicionar Sitio Web:
El usuario debe especificar en que idioma desea adicionar la información referente al sitio Web.

El sistema debe permitir al usuario introducir la información referente al sitio Web:
- Temática.
- Dirección.
- Título.

- Descripción.

R13.2 Listar Sitio Web:
El sistema debe permitir al usuario especificar en que idioma desea listar los sitios Web.

El sistema debe permitir al usuario verificar la información referente a los sitios Web:
- Título.
- Descripción.

R13.3 Modificar Sitio Web:
El sistema debe permitir al usuario especificar en que idioma desea modificar la información del sitio Web.

El sistema debe permitir al usuario modificar la información referente al sitio Web:
- Categoría.
- Dirección.
- Título.
- Descripción.

R13.4 Eliminar Sitio Web:
El sistema debe eliminar toda la información referente al sitio Web que el usuario desee suprimir.

R14 Gestionar Solicitudes del Servicio de Lavandería.
R14.1 Listar Solicitudes del Servicio de Lavandería:
El sistema debe permitir al usuario verificar la información referente a las solicitudes del servicio de lavandería:
- Nombre del solicitante.
- Tipo de servicio (lavado y/o planchado).
- Fecha de la solicitud.
- Zona residencial.
- Residencia.
- Habitación.
- Estado.

R14.2 Eliminar Solicitud del Servicio de Lavandería:
El sistema debe eliminar toda la información referente a la solicitud del servicio de lavandería que el usuario desee suprimir.

R15 Gestionar Encuestas.
R15.1 Adicionar Encuesta:
El sistema debe permitir al usuario especificar en que idioma desea adicionar la información referente a la encuesta.

El sistema debe permitir al usuario introducir la información referente a la encuesta:
- Fecha de publicación.
- Fecha de caducidad.
- Cantidad de respuestas.
- Pregunta.
- Respuestas.

R15.2 Listar Encuesta:
El sistema debe permitir al usuario especificar en que idioma desea listar las encuestas.

El sistema debe permitir al usuario verificar la información referente a las encuestas:
- Fecha de publicación.
- Fecha de caducidad.
- Pregunta.
- Respuestas.
- Porciento de votos por respuesta.

R15.3 Modificar Encuesta:
El sistema debe permitir al usuario especificar en que idioma desea modificar la información de la encuesta.

El sistema debe permitir al usuario modificar la información referente a la encuesta:
- Fecha de publicación.
- Fecha de caducidad.
- Cantidad de respuestas.
- Pregunta.
- Respuestas.

R15.4 Eliminar Encuesta:
El sistema debe eliminar toda la información referente a la encuesta que el usuario desee suprimir.

R15.5 Visualizar Encuesta:
El sistema debe permitir activar y desactivar la encuesta, es decir, si debe estar disponible o no para el usuario.

R15.6 Adicionar Respuesta:
El usuario debe especificar en que idioma desea adicionar la información referente a la respuesta.

El sistema debe permitir al usuario introducir la información referente a la respuesta:
- Encuesta.
- Respuesta.

R15.7 Eliminar Respuesta:
El sistema debe eliminar toda la información referente a la respuesta que el usuario desee suprimir.

R15.8 Restablecer Votos:
El sistema debe permitir colocar en 0% los votos de las respuestas de la encuesta.

R16 Gestionar Residencia.

R16.1 Adicionar Residencia:
El sistema debe permitir al usuario especificar en que idioma desea adicionar la información referente a la residencia.

El sistema debe permitir al usuario introducir la información referente a la residencia:
- Imagen.
- Número de IP.
- Zona.
- Número.
- Nivel.

- Nombre.
- Reseña Histórica.
- Interiores.
- Dirección.
- Descripción de la imagen.

R16.2 Listar Residencias:
El sistema debe permitir al usuario especificar en que idioma desea listar las residencias.

El sistema debe permitir al usuario verificar la información referente a las residencias:
- Número de IP.
- Nivel.
- Zona.
- Número.

R16.3 Modificar Residencia:
El sistema debe permitir al usuario especificar en que idioma desea modificar la información de la residencia.

El sistema debe permitir al usuario modificar la información referente a la residencia:
- Imagen.
- Número de IP.
- Zona.
- Número.
- Nivel.
- Nombre.
- Reseña Histórica.
- Interiores.
- Dirección.
- Descripción de la imagen.

R16.4 Eliminar Residencia:
El sistema debe ser capaz de eliminar toda la información referente a la residencia que el usuario desee suprimir.

R17 Gestionar Obra de Arte.
R17.1 Adicionar Obra de Arte:
El sistema debe permitir al usuario especificar en que idioma desea adicionar la información referente a la obra de arte.

El sistema debe permitir al usuario introducir la información referente a la obra de arte:
- Residencia de la obra de arte
- Imagen de la obra.
- Título de la obra.
- Fecha de la obra.
- Origen de la obra.
- Autores de la obra.
- Descripción de la obra.
- Manifestación de la obra.
- Denominación de la obra.
- Época de la obra.
- Técnica de la obra.

- Medidas de la obra.
- Materiales de la obra.
- Soporte de la obra.
- Color de la obra.
- Estado de la obra.
- Valor de la obra.
- Grado de valor de la obra.
- Tasación de la obra.
- Ubicación actual de la obra.
- Catalogador de la obra.
- Número de inventario de la obra.
- Temática de la obra.
- Lugar de la firma de la obra.
- Escuela del autor de la obra.
- Casa editora de la obra.
- Forma de la obra.

R17.2 Listar Obra de Arte:
El sistema debe permitir al usuario especificar en que idioma desea listar las obras de arte.

El sistema debe permitir al usuario verificar la información referente a la obra de arte:
- Título.
- Residencia.
- Zona.

R17.3 Modificar Obra de Arte:
El usuario debe especificar en que idioma desea modificar la información de la obra de arte.

El sistema debe permitir al usuario modificar la información referente a la obra de arte:
- Residencia de la obra de arte
- Imagen de la obra.
- Título de la obra.
- Fecha de la obra.
- Origen de la obra.
- Autores de la obra.
- Descripción de la obra.
- Manifestación de la obra.
- Denominación de la obra.
- Época de la obra.
- Técnica de la obra.
- Medidas de la obra.
- Materiales de la obra.
- Soporte de la obra.
- Color de la obra.
- Estado de la obra.
- Valor de la obra.
- Grado de valor de la obra.
- Tasación de la obra.
- Ubicación actual de la obra.
- Catalogador de la obra.
- Número de inventario de la obra.

- Temática de la obra.
- Lugar de la firma de la obra.
- Escuela del autor de la obra.
- Casa editora de la obra.
- Forma de la obra.

R17.4 Eliminar Obra de arte:
El sistema debe eliminar toda la información referente a la obra de arte que el usuario desee suprimir.

R18 Gestionar Usuario.
R18.1 Adicionar Usuarios:
El sistema debe permitir al usuario introducir la información:
- Usuario.
- Contraseña.
- Nombre.
- Apellidos.
- Correo.
- Departamento.

R18.2 Listar Usuarios:
El sistema debe permitir al usuario verificar la información:
- Nombre.
- Apellidos.
- Departamento.
- Módulos a los que le es permitido acceder.

R18.3 Modificar Usuarios:
El sistema debe permitir al usuario modificar la información:
- Usuario.
- Contraseña.
- Nombre.
- Apellidos.
- Correo.
- Departamento.
- Módulos a los que le es permitido acceder.

R18.4 Eliminar Usuario:
El sistema debe eliminar toda la información referente al usuario que se desee suprimir.

R19 Gestionar Rol.
R19.1 Adicionar Rol:
El sistema debe permitir introducir un nuevo rol al sistema, dando la posibilidad de introducir los datos:
- Nombre del rol.
- Módulo.
- Permisos.

R19.2 Listar Rol:
El sistema debe permitir al usuario verificar la información:
- Nombre del rol.
- Módulo.
- Permisos.

R19.3 Modificar Rol:

El sistema debe permitir al usuario modificar la información:

- Nombre del rol.
- Módulo.
- Permisos.

R19.4 Eliminar Rol:

El sistema debe permitir eliminar un rol.

R20 Gestionar Personal.

R20.1 Adicionar Personal:

El sistema debe permitir al usuario especificar en que idioma desea adicionar la información referente al personal.

El sistema debe permitir al usuario introducir la información referente al personal:

- Nombre.
- Categoría.
- Zona.
- Residencia.
- Foto.
- Descripción de la foto.

R20.2 Listar Personal:

El sistema debe permitir al usuario especificar en que idioma desea listar el personal.

El sistema debe permitir al usuario verificar la información referente al personal:

- Nombre.
- Categoría.
- Ubicación.

R20.3 Modificar Personal:

El sistema debe permitir al usuario especificar en que idioma desea modificar la información del personal.

El sistema debe permitir al usuario modificar la información referente al personal:

- Nombre.
- Categoría.
- Zona.
- Residencia.
- Foto.
- Descripción de la foto.

R20.4 Eliminar Personal:

El sistema debe eliminar toda la información referente al personal que el usuario desee suprimir.

R21 Gestionar Imagen de Galería.

R21.1 Adicionar Imagen de Galería:

El sistema debe permitir al usuario especificar en que idioma desea adicionar la información referente a la imagen.

El sistema debe permitir al usuario introducir la información referente a la imagen:

- Imagen.
- Categoría.
- Descripción.

R21.2 Listar las Imágenes de la Galería:
El sistema debe permitir al usuario especificar en que idioma desea listar las imágenes de la galería.

El sistema debe permitir al usuario verificar la información referente a la imagen:
- Descripción.
- Categoría.

R21.3 Modificar Imagen de Galería:
El sistema debe permitir al usuario especificar en que idioma desea modificar la información de la imagen.

El sistema debe permitir al usuario modificar la información referente a la imagen:
- Imagen.
- Categoría.
- Descripción.

R21.4 Eliminar Imagen de la Galería:
El sistema eliminar toda la información referente a la imagen que el usuario desee suprimir.

R22 Gestionar Opinión:
R22.2 Listar Opiniones:
El sistema debe permitir al usuario verificar la información referente a la opinión:
- Servicio del que se opina.
- Fecha.
- Nombre del que opina.
- Residencia desde la que se hace la opinión.
- Opinión.

R22.4 Eliminar Opinión:
El sistema debe eliminar toda la información referente a la opinión que el usuario desee suprimir.

2.6.2 Requerimientos No Funcionales.

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable.

Apariencia o interfaz externa.

- Evitar recargar las páginas con textos, imágenes o gráficos.

- Establecer mecanismos de barrido visual para el contenido de la página, distribuyendo los elementos de información y navegación según su importancia en zonas de mayor o menor jerarquía visual. Las zonas superiores de la interfaz poseen más jerarquía visual que las inferiores.

- Utilizar colores sobrios y ofrecer suficiente contraste entre texto y fondo para no dificultar la lectura.

- Mantener una coherencia y estilo común entre todas las páginas.

Usabilidad.

- La Intranet debe estar orientada a usuarios con conocimientos básicos de informática y trabajo con la Web.

- Ofrecer señalizaciones que agilicen el aprendizaje del usuario a trabajar con el sistema.

Rendimiento.

- Debe contar con un soporte de hardware que garantice un ágil procesamiento de los datos y un tiempo de respuestas rápidas.

Soporte.

- La Universidad de las Ciencias Informáticas garantiza la instalación del sistema en los ordenadores del cliente.

- La Universidad de las Ciencias Informáticas garantiza el soporte y mantenimiento del sistema siempre que sea solicitado por el cliente.

Portabilidad.

- El sistema debe ser multiplataforma.

Seguridad.

- Solicitar a todo usuario su identificación antes de que pueda ejecutar cualquier acción sobre la Intranet.

- Crear diferentes cuentas de usuario y asignarle a cada uno los permisos pertinentes.

- Mostrar a cada usuario solo las funcionalidades del sistema sobre las cuales tiene permiso de acceso.

- Ofrecer mensajes de verificación antes de ejecutar acciones irreversibles (eliminaciones de datos).

- El servidor donde se encuentre instalado el sistema debe estar ubicado en un local protegido contra el hurto y los desastres naturales.

Políticos-culturales.

- La Intranet estará concebida para Las Residencias de Protocolo del Consejo de Estado del la República de Cuba y deberá tributar a la política exterior cubana y estar acorde con los principios éticos de la Revolución.

Legales.

- La Intranet fue desarrollada con el lenguaje de programación PHP perteneciente a la licencia de software libre GNU/GPL.

Confiabilidad.

- La máquina servidora donde se encuentre instalada el sistema debe estar conectada a una fuente de almacenamiento de energía que garantice su funcionamiento cuando ocurran fallos del fluido eléctrico.

Software.

- Navegador compatible o superior con Internet Explorer 4, o Netscape Navegador.
- Apache 2.0.58
- MySQL 5.0.15
- PHP 5.1.4

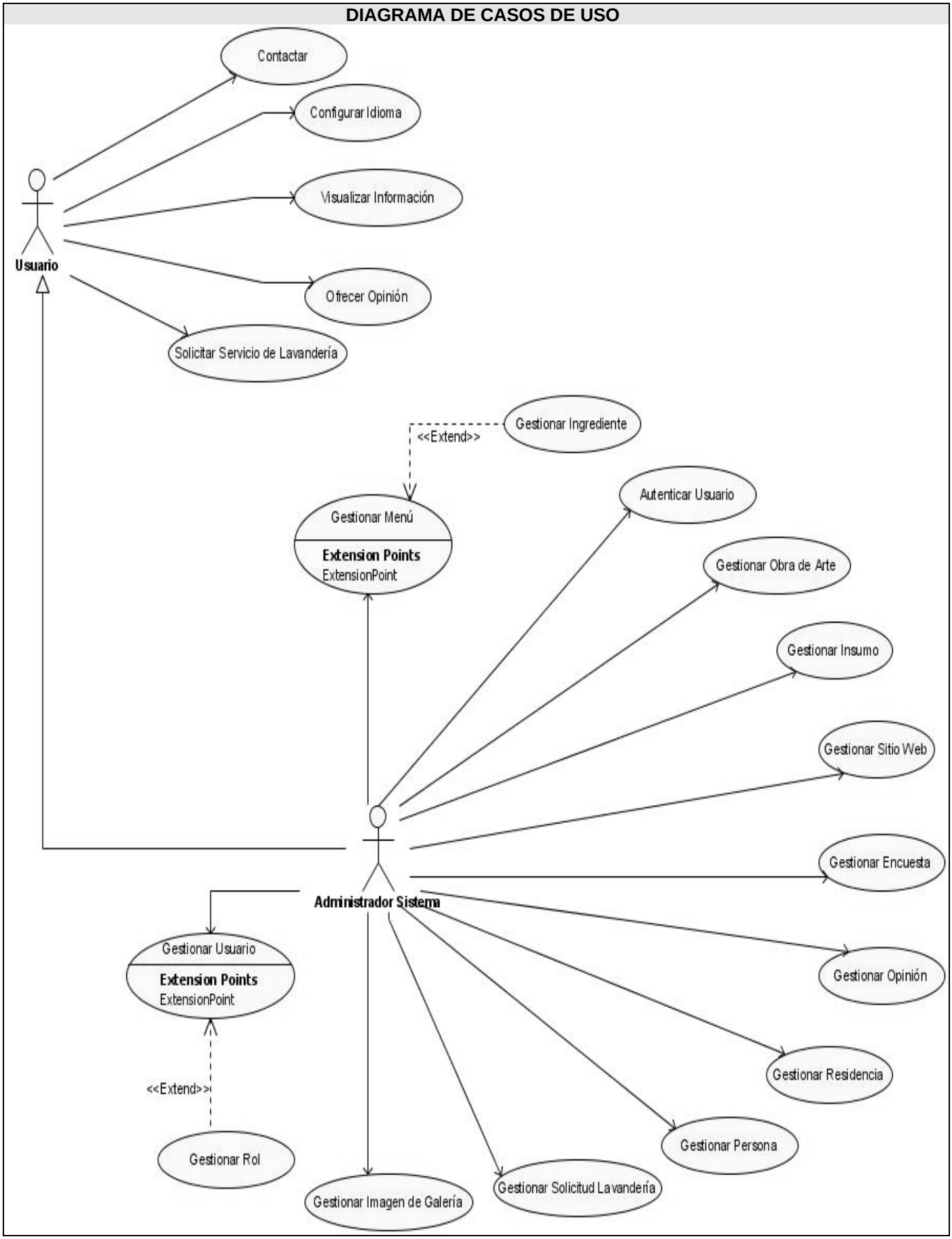
Ayuda y documentación en línea.

- Cada funcionalidad debe tener una ayuda, en la que se dará una explicación detallada del trabajo con la misma.

2.7 Definición de los Actores del Sistema.

Actores	Justificación
Usuario	Es el encargado de seleccionar el idioma en que se mostrará el contenido de la Intranet, establecer contacto a través de la Intranet con la entidad, realizar encuestas, ofrecer su opinión, solicitar el servicio de lavandería y obtener la información de la Intranet.
Administrador del Sistema	Es el encargado de gestionar el rol de los usuarios, los usuarios, el personal, el menú, los ingredientes, las bebidas, los vinos, los cocteles, los sitios Web, las obras de arte, la información sobre las residencias, las encuestas, las opiniones, las imágenes de la galería y las solicitudes de lavandería.

DIAGRAMA DE CASOS DE USO



2.8 Descripciones de los Casos de Uso del Sistema.

Caso de uso		
CU-1	Configurar Idioma.	
Propósito	Permitir al usuario acceder al contenido de la Intranet en el idioma que desee.	
Actores	Usuario	
Resumen: El caso de uso se inicia cuando el usuario accede a la opción que el sistema ofrece para cambiar el idioma en que se muestra el contenido, selecciona el idioma que desea e inmediatamente el sistema muestra el contenido traducido al idioma que fue seleccionado.		
Referencias	R1	
Acción del actor		**Respuesta del sistema**
1. El usuario selecciona la opción para cambiar el idioma.		1.1 El sistema le muestra la lista de idiomas en que puede ser mostrado el contenido.
2. El usuario selecciona el idioma que desea.		2.1 El sistema muestra el contenido en el idioma que el usuario seleccionó.
Flujo alternativo		
Acción del actor		**Respuesta del sistema**
Puntos de extensión.		

Caso de uso		
CU-2	Solicitar Servicio de Lavandería.	
Propósito	Permitir que los usuarios soliciten el servicio de lavandería.	
Actores	Usuario	
Resumen: El caso de uso se inicia cuando el usuario accede al vínculo solicitud de lavandería, el sistema muestra un formulario, el usuario llena dicho formulario enviando la solicitud.		
Referencias	R4	
Acción del actor		**Respuesta del sistema**
1. El usuario accede al vínculo de solicitud de lavandería.		1.1 El sistema muestra un formulario con el nombre, habitación, servicio y el botón enviar solicitud.
2. El usuario llena el formulario.		2.1 El sistema muestra un cartel informando que la solicitud ha sido enviada correctamente.
Flujo alternativo		
Acción del actor		**Respuesta del sistema**
Línea 2		
2. El usuario introduce los datos incorrectamente.		2.1 El sistema muestra un mensaje que anuncia que ha habido datos introducidos incorrectamente. 2.2 El sistema vuelve a solicitar los datos al usuario.
Puntos de extensión.		

Caso de uso	
CU-3	Ofrecer Opinión.
Propósito	Permitir al usuario ofrecer su opinión.
Actores	Usuario

Resumen: Es aquí donde se registran las opiniones de los usuarios y sus respuestas a las encuestas.	
Referencias	R6
Acción del actor	**Respuesta del sistema**
1. El usuario desea ofrecer su opinión sobre un servicio o contestar una encuesta.	El sistema ejecuta alguna de las siguientes acciones: a) Si decide ofrecer su opinión sobre un servicio ir a la sección "Ofrecer Opinión". b) Si decide contestar una encuesta ir a la sección "Contestar Encuesta".
Sección "Ofrecer Opinión"	
Acción del actor	**Respuesta del sistema**
1. El usuario decide ofrecer su opinión.	1.1 El sistema muestra un formulario con los datos que se necesitan introducir para ofrecer la opinión.
2. El usuario introduce los datos correctamente.	2.1 El sistema guarda los datos y muestra un mensaje anunciando que la opinión ha sido ofrecida correctamente.
Flujo alternativo	
Línea 2	
2. El usuario introduce los datos incorrectamente.	2.1 El sistema muestra un mensaje que anuncia que ha habido datos introducidos incorrectamente. 2.2 El sistema vuelve a solicitar los datos al usuario.
Sección "Contestar Encuesta"	
Acción del actor	**Respuesta del sistema**
1. El usuario decide contestar una encuesta.	1.1 El sistema muestra un formulario con los datos que se necesitan introducir para realizar la encuesta.
2. El usuario introduce los datos correctamente.	2.1 El sistema guarda los datos y muestra un mensaje anunciando que la encuesta ha sido realizada correctamente.
Flujo alternativo	
Línea 2	
2. El usuario introduce los datos incorrectamente.	2.1 El sistema muestra un mensaje que anuncia que ha habido datos introducidos incorrectamente. 2.2 El sistema vuelve a solicitar los datos al usuario.
Puntos de extensión.	

Caso de uso		
CU-4	Contactar	
Propósito	Permitir al usuario poder contactar con el personal de las Residencias de Protocolo mediante una dirección de correo electrónico.	
Actores	Usuario	
Resumen: El caso de uso se inicia cuando el usuario accede al vínculo del contacto. El sistema le muestra un cliente de correo electrónico (con la dirección del destinatario predefinida) donde el usuario introduce su mensaje, luego el sistema envía el mensaje al destinatario.		
Referencias	R2	
Acción del actor		**Respuesta del sistema**
1. El usuario accede al vínculo del contacto.		1.1 El sistema muestra un cliente de correo electrónico (con la dirección del destinatario predefinida).
2. El usuario introduce el mensaje.		2.1 El sistema envía el mensaje al destinatario.
Flujo alternativo		
Acción del actor		**Respuesta del sistema**
Puntos de extensión.		

Caso de uso		
CU-5	Autenticar Administrador del Sistema.	
Propósito	Permitir autenticarse.	
Actores	Administrador del Sistema.	
Resumen: El caso de uso se inicia cuando el Administrador del Sistema introduce los datos que se le piden para acceder a la Intranet, estos se verifican y se le da o no el permiso de acceso.		
Referencias	R7	
Acción del actor		**Respuesta del sistema**
1. El Administrador del sistema introduce usuario y contraseña.		1.1 Es sistema encripta la contraseña. 1.2 El sistema verifica que estén correctos el usuario y la contraseña. 1.3 En caso de ser correcto se le asignan los permisos.
Flujo alternativo		
Acción del actor		**Respuesta del sistema**
Línea 1.3		
		1.3 En caso de no existir se envía un mensaje de aviso y se le niega el acceso.
Puntos de extensión.		

Caso de uso	
CU-6	Gestionar Menú.
Propósito	Permitir registrar y listar los datos acerca del menú del almuerzo y la cena de cada día de la semana.
Actores	Administrador del Sistema.
Resumen: Es aquí donde se registran y listan los datos del menú del almuerzo y la cena de cada día de la semana.	

Referencias	R8, R8.1, R8.2, R8.3
Acciones del Actor	**Respuesta del Sistema**
1. El administrador del sistema necesita registrar y listar el menú de almuerzo y cena.	1.1 El sistema ejecuta alguna de las siguientes acciones: a) Si decide registrar un nuevo menú de almuerzo ir a la sección “Confeccionar Menú de Almuerzo”. b) Si decide registrar un nuevo menú de la cena ir a la sección “Confeccionar Menú de Cena”. c) Si decide listar los datos del menú, ir a la sección "Listar Menú". d) Si decide gestionar un nuevo ingrediente ir a la sección “Gestionar Ingredientes”.
Sección “Confeccionar Menú del Almuerzo”	
1. El administrador del sistema selecciona el idioma en que desea introducir los datos.	
2. El administrador del sistema entra los datos del Menú para realizar su registro en la aplicación.	2.1 El sistema verifica que los campos de la fecha, día de la semana, entrante, guarnición, plato fuerte y postre estén llenos. 2.2 El menú del almuerzo de esa fecha es actualizado y se almacena en el sistema. 2.3 Se muestra un mensaje informándosele al administrador que ya ha sido efectuado el registro del menú y finaliza el caso de uso.
Flujo alternativo	
Línea 2.2	
	2.2 El sistema muestra un mensaje anunciando que se han efectuado errores cuando se introdujeron los datos.
Línea 2.3	
	2.3 El sistema muestra al administrador la interfase donde debe proporcionar los datos nuevamente.
Sección “Confeccionar Menú de Cena”	
1. El administrador del sistema selecciona el idioma en que desea introducir los datos.	

2. El administrador del sistema entra los datos del Menú para realizar su registro en la aplicación.	2.1 El sistema verifica que los campos de la fecha, día de la semana, volante, guarnición, plato fuerte y postre estén llenos. 2.2 El menú de la cena de asa fecha es actualizado y se almacena en el sistema. 2.3 Se muestra un mensaje informándosele al administrador que ya ha sido efectuado el registro del menú y finaliza el caso de uso.
Flujo alternativo	
Línea 2.2	
	2.2 El sistema muestra un mensaje anunciando que se han efectuado errores cuando se introdujeron los datos.
Línea 2.3	
	2.3 El sistema muestra al administrador la interfase donde debe proporcionar los datos nuevamente.
Sección "Listar Menú"	
1. El administrador del sistema selecciona el idioma en que desea listar los datos del menú. **Sección "Gestionar Ingredientes"**	1.1 El sistema muestra un listado con los datos del menú del almuerzo y la cena de cada día de la semana organizada por tipo de plato en el idioma que fue seleccionado.
Ver Caso de Uso extendido: Gestionar Ingredientes.	
Puntos de extensión.	

Caso de uso		
CU-7	Gestionar Solicitudes del Servicio de Lavandería.	
Propósito	Permitir listar y eliminar las solicitudes de la lavandería.	
Actores	Administrador del Sistema.	
Resumen: Es aquí donde se listan y eliminan los datos de las solicitudes de lavandería.		
Referencias	R14, R14.1, R14.2.	
Acción del actor		**Respuesta del sistema**
1. El administrador del sistema necesita listar o eliminar los datos de una solicitud de la lavandería.		1.1 El sistema ejecuta alguna de las siguientes acciones: a) Si decide listar una solicitud, ir a la sección "Listar Solicitud". b) Si decide eliminar una solicitud, ir a la sección "Eliminar Solicitud".
Sección "Listar Solicitud"		
1. El administrador del sistema selecciona el idioma en que desea listar los datos de la solicitud.		1.1 El sistema muestra un listado con los datos nombre del solicitante, tipo de servicio (lavado y/o planchado), fecha de la solicitud, zona residencial, residencia, habitación, estado de cada solicitud que se encuentra

	en el sistema.
Sección "Eliminar Solicitud"	
1. El administrador del sistema afirma que la solicitud fue realizada.	1.1 El sistema pide confirmación. 1.2 El sistema elimina la solicitud.
Puntos de extensión.	

Caso de uso		
CU-8	Gestionar Encuesta.	
Propósito	Permitir adicionar, listar, modificar y eliminar encuestas.	
Actores	Administrador	
Resumen: Es aquí donde se registran, listan, eliminan y modifican los datos de las encuestas, así como también, se adicionan y eliminan las respuestas de las encuestas y se restablecen los resultados de los votos.		
Referencias	R15, R15.1, R15.2. R15.3, R15.4, R15.5, R15.6, R15.7, R15.8.	
Acción del actor		**Respuesta del sistema**
1. El administrador del sistema necesita registrar, listar, modificar o eliminar los datos de una encuesta. Puede también adicionar y eliminar la respuesta de las encuestas o restablecer los resultados de los votos.		1.1 El sistema ejecuta alguna de las siguientes acciones: a) Si decide registrar una encuesta, ir a la sección "Adicionar Encuesta." b) Si decide listar una encuesta, ir a la sección "Listar Encuesta". c) Si decide actualizar las características de una encuesta, ir a la sección "Editar Encuesta". d) Si decide eliminar una encuesta, ir a la sección "Eliminar Encuesta". e) Si decide registrar una respuesta, ir a la sección "Adicionar Respuesta". f) Si decide eliminar una respuesta, ir a la sección "Eliminar Respuesta". g) Si decide activar o desactivar una encuesta, ir a la sección "Visualizar Encuesta". h) Si decide restablecer una respuesta, ir a la sección "Restablecer Voto".
Sección "Adicionar Encuesta".		
1. El administrador del sistema selecciona el idioma en que desea introducir los datos.		
2. El administrador del sistema entra los datos de la encuesta para realizar su registro en la aplicación.		2.1 El sistema verifica que los campos de fecha de publicación, fecha de caducidad, cantidad de respuestas, pregunta y respuestas de la encuesta estén llenos. 2.2 El sistema verifica que esta encuesta no exista. 2.3 La encuesta se almacena en el sistema. 2.4 Se muestra un mensaje informándosele al administrador que ya ha sido efectuado el registro de la encuesta y finaliza el caso de

	uso.
Flujo alternativo	
Línea 2.1	
	2.1 Se emite un mensaje para que llene los campos obligatorios.
Línea 2.2	
	2.2 Si la encuesta existe se muestra un mensaje informativo y finaliza el caso de uso.
Sección "Listar Encuesta"	
1. El administrador del sistema selecciona el idioma en que desea listar los datos de la encuesta.	1.1 El sistema muestra un listado con los datos fecha de publicación, fecha de caducidad, pregunta, respuestas y porciento de votos por respuesta de cada encuesta que se encuentra en el sistema.
Sección "Modificar Encuesta"	
1. El administrador del sistema selecciona la encuesta a modificar.	1.1 El sistema brinda la posibilidad de modificar los datos.
2. El administrador del sistema realiza las actualizaciones deseadas.	2.1 Se verifica que los campos obligatorios estén llenos. 2.2 Se actualiza la información y finaliza el caso de uso.
Flujo alternativo	
Línea 2.1	
	2.1 Se emite un mensaje para que llene los campos obligatorios.
Sección "Eliminar Encuesta"	
2. El administrador del sistema selecciona la encuesta que desea eliminar.	2.1 Es sistema pide confirmación. 2.2 El sistema elimina la encuesta.
Sección "Visualizar Encuesta"	
1. El administrador del sistema selecciona la opción activar.	1.1 El sistema permite que la encuesta esté disponible para el usuario.
Flujo alternativo	
Línea 1	
1. El administrador del sistema selecciona la opción desactivar.	1.1 El sistema permite que la encuesta no esté disponible para el usuario.
Sección "Adicionar Respuestas"	
1. El administrador del sistema selecciona el idioma en que desea introducir los datos.	1.1 El sistema verifica que el campo de respuesta esté lleno. 2.2 El sistema verifica que esta respuesta no exista. 2.3 La respuesta se almacena en el sistema. 2.4 Se muestra un mensaje informándosele al administrador que ya ha sido efectuado el registro de la respuesta y finaliza el caso de uso.

Flujo alternativo	
Línea 2.1	
	2.1 Se emite un mensaje para que llene los campos obligatorios.
Línea 2.2	
	2.2 Si la respuesta existe se muestra un mensaje informativo y finaliza el caso de uso.
Sección "Eliminar Respuestas"	
1. El administrador acepta eliminar la respuesta.	1.1 El sistema pide confirmación. 1.2 El sistema elimina la respuesta.
Sección "Restablecer Votos"	
1. El administrador del sistema selecciona la opción restablecer votos.	1.1 El sistema pide confirmación. 1.2 El sistema coloca en 0% los votos de las respuestas de la encuesta.
Puntos de extensión.	

Caso de uso		
CU-9	Gestionar Usuario.	
Propósito	Permitir adicionar, listar, modificar y eliminar usuarios.	
Actores	Administrador del Sistema.	
Resumen: Es aquí donde se registran, listan, eliminan y modifican los datos de los usuarios.		
Referencias	R18, R18.1, R18.2, R18.3, R18.4.	
Acción del actor		**Respuesta del sistema**
1. El administrador del sistema necesita registrar, listar, modificar o eliminar los datos de un usuario.		1.1 El sistema ejecuta alguna de las siguientes acciones: a) Si decide registrar un usuario, ir a la sección "Adicionar Usuario.". b) Si decide listar un usuario, ir a la sección "Listar Usuario". c) Si decide actualizar las características de un usuario, ir a la sección "Editar Usuario". d) Si decide eliminar un usuario, ir a la sección "Eliminar Usuario".
Sección "Adicionar Usuario".		
1. El administrador del sistema entra los datos del usuario para realizar su registro en la aplicación.		1.1 El sistema verifica que los campos de usuario, contraseña, nombre, apellidos, correo y departamento del usuario estén llenos. 1.2 El sistema verifica que este usuario
		no exista. 1.3 El usuario se almacena en el sistema. 1.4 Se muestra un mensaje informándosele al administrador que ya ha sido efectuado el registro del usuario y finaliza el caso de uso.
Flujo alternativo		
Línea .1.1		

	1.1 Se emite un mensaje para que llene los campos obligatorios.
Línea 1.3	
	1.3 Si el usuario existe se muestra un mensaje informativo y finaliza el caso de uso.
Sección "Listar Usuario"	
1. El administrador del sistema decide listar los datos de los usuarios.	1.1 El sistema muestra un listado con los datos nombre, apellidos, departamento y módulos a los que le es permitido acceder de cada usuario que se encuentra en el sistema.
Sección "Editar Usuario"	
1. El administrador del sistema selecciona el usuario a modificar.	1.1 El sistema brinda la posibilidad de modificar los datos.
2. El administrador del sistema realiza las actualizaciones deseadas.	2.1 Se verifica que los campos obligatorios estén llenos. 2.2 Se actualiza la información y finaliza el caso de uso.
Flujo alternativo	
Línea 2.1	
	2.1 Se emite un mensaje para que llene los campos obligatorios.
Sección "Eliminar Usuario"	
2. El administrador de sistema selecciona el usuario que desea eliminar.	2.1 El sistema pide confirmación. 2.2 El sistema elimina el usuario.
Puntos de extensión.	

Caso de uso		
CU-10	Gestionar Rol.	
Propósito	Permitir adicionar, listar, modificar y eliminar los roles que se le asignan a los usuarios.	
Actores	Administrador del Sistema.	
Resumen: Es aquí donde se registran, listan, eliminan y modifican los datos de los roles que se le asignan a los usuarios.		
Referencias	R19, R19.1, R19.2, R19.3, R19.4.	
Acción del actor		**Respuesta del sistema**
1. El administrador del sistema necesita registrar, listar, modificar o eliminar los datos de un rol.		1.1 El sistema ejecuta alguna de las siguientes acciones: a) Si decide registrar un rol, ir a la sección "Adicionar Rol.". b) Si decide listar un rol, ir a la sección "Listar Rol". c) Si decide actualizar las características de un rol, ir a la sección "Editar Rol".

	d) Si decide eliminar un usuario, ir a la sección "Eliminar Rol".
Sección "Adicionar Rol".	
1. El administrador del sistema entra los datos del rol para realizar su registro en la aplicación.	1.1 El sistema verifica que el campo nombre del rol este lleno. 1.2 El sistema verifica que este rol no exista. 1.3 El rol se almacena en el sistema. 1.4 Se muestra un mensaje informándosele al administrador que ya ha sido efectuado el registro del rol y finaliza el caso de uso.
Sección "Listar Rol"	
	1.1 El sistema muestra un listado con el nombre de cada rol que se encuentra en el sistema.
Sección "Modificar Rol"	
1. El administrador del sistema realiza las actualizaciones deseadas.	1.1 Se verifica que los campos obligatorios estén llenos. 1.2 Se actualiza la información y finaliza el caso de uso.
Sección "Eliminar Rol"	
1. El administrador selecciona el rol que desea eliminar.	1.1 El sistema pide confirmación. 1.2 El sistema elimina el rol.
Puntos de extensión.	

Caso de uso		
CU-11	Visualizar Información.	
Actores	Usuario	
Propósito	Mostrar al usuario la información de la Intranet.	
Resumen: Es aquí donde se muestra la información relacionada con las residencias, los servicios (alojamiento, lavandería, gastronomía, protocolo, informática, televisión y trasporte), los sitios Web cubanos, el sitial histórico, obras de arte y las imágenes de la galería.		
Referencias	R3, R3.1, R3.2, R3.3, R3.4, R3.5, R3.6, R3.7, R3.8, R3.9.	
Acción del actor		**Respuesta del sistema**
1. El usuario necesita obtener una información.		1.1 El sistema ejecuta alguna de las siguientes acciones: a) Si decide ver la historia de la residencia, ir a la sección "Historia". b) Si decide ver la información de las residencias, ir a la sección "Residencias". c) Si decide ver la información de los servicios, ir a la sección "Servicios". d) Si decide ver la información de sitios Web cubanos, ir a la sección "Sitios Web Cubanos". e) Si decide ver la información del sitial histórico, ir a la sección "Sitial Histórico". f) Si decide ver la información de las

	obras de arte, ir a la sección “Obras de Arte”. g) Si decide ver imágenes relacionadas con alguno de los temas de la Intranet, ir a la sección “Imágenes de la Galería”. h) Si decide ver el mapa de navegación de la Intranet, ir a la sección “Mapa de navegación”.
Sección “Historia”	
1. El usuario accede al vínculo de historia.	1.1 El sistema muestra la historia de la residencia.
Sección “Residencias”	
1. El usuario accede al vínculo de residencias.	1.1 El sistema muestra todas las opciones: Interiores, Servicios, Personal, Mapa, Zona 1, Zona 2 y Zona Varadero.
2. El usuario selecciona una de las opciones: Interiores, Servicios, Personal o Mapa.	2.1 El sistema muestra el contenido de la opción seleccionada.
Flujo alternativo	
Acción del actor	**Respuesta del sistema**
Línea 2	
1. El usuario selecciona una de las zonas.	1.1 El sistema muestra un listado con las residencias correspondiente a dicha zona.
2. El usuario selecciona una residencia.	2.1 El sistema muestra la información de la residencia.
Sección “Servicios”	
1. El usuario accede al vínculo de los servicios.	1.1 El sistema muestra todos los servicios de los que guarda información: Alojamiento, Lavandería, Televisión por Cable, Transporte, Informática y Gastronomía.
2. El usuario selecciona uno de los servicios de: Alojamiento, Televisión por Cable, Informática o Transporte.	2.1 El sistema muestra el contenido del servicio seleccionado.
Flujo alternativo	
Acción del actor	**Respuesta del sistema**
Línea 2	
2. El usuario accede al vínculo del servicio de gastronomía.	2.1 El sistema muestra la información relacionada con este servicio y las clasificaciones de contenido: Menú, Bebidas, Vinos y Cocteles.
3. El usuario selecciona una de las clasificaciones: Menú, Bebidas, Vinos o Cocteles.	3.1 El sistema muestra el contenido correspondiente a dicha selección.
Sección “Sitios Web Cubanos”	
1. El usuario accede al vínculo de los sitios Web cubanos.	1.1 El sistema muestra un listado de temáticas de sitios Web.
2. El usuario selecciona una temática.	2.1 El sistema muestra un listado de vínculos a sitios de esa temática.
Sección “Sitial Histórico”	
1. El usuario accede al vínculo del sitial histórico.	1.1 El sistema muestra las opciones: Sitial y Celia.

2. El usuario selecciona una opción.	2.1 El sistema muestra la información de la opción seleccionada.
Sección "Obras de Arte"	
1. El usuario accede al vínculo de las obras de arte.	1.1 El sistema muestra un listado de las obras de arte que se encuentran en la residencia.
2. El usuario selecciona una obra de arte.	2.1 El sistema muestra la información referente a la obra de arte que el usuario seleccionó.
Sección "Imágenes de la Galería"	
1. El usuario accede al vínculo de la galería.	1.1 El sistema muestra una lista con diferentes opciones.
2. El usuario selecciona una opción.	2.1 El sistema muestra una lista de imágenes referentes a dicha opción.
Sección "Mapa de Navegación"	
1. El usuario accede al vínculo del mapa.	1.1 El sistema muestra una lista de vínculos, que representan todas las páginas de la Intranet a las que el usuario puede acceder.
2. El usuario selecciona un vínculo.	2.1 El sistema muestra la información del vínculo seleccionando.
Puntos de extensión.	

Caso de uso		
CU-12	Gestionar Residencia.	
Propósito	Permitir registrar, listar, eliminar y modificar datos acerca de las Residencias.	
Actores	Administrador del Sistema.	
Resumen: Es aquí donde se registran, listan, eliminan y modifican los datos de las Residencias.		
Referencias	R16, R16.1, R16.2, R16.3, R16.4.	
Curso Normal de los Eventos		
Acciones del Actor		**Respuesta del Sistema**
1. El administrador del sistema necesita registrar, listar, eliminar o modificar los datos de una Residencia.		1.1 El sistema ejecuta alguna de las siguientes acciones: a) Si decide registrar una residencia ir a la sección "Adicionar Residencia". b) Si decide listar las residencias ir a la sección "Listar Residencias". c) Si decide actualizar las características de una residencia, ir a la sección "Editar Residencia". d) Si decide eliminar una residencia, ir a la sección "Eliminar Residencia".
Sección "Adicionar Residencia"		
Acciones del Actor		**Respuesta del Sistema**
1. El administrador del sistema selecciona el idioma en que desea registrar los datos de la residencia.		
2. El administrador del sistema entra los datos de la		2.1 El sistema verifica que los campos

residencia para realizar su registro en la aplicación.	de imagen, número de ip, zona, número, nivel, nombre, reseña, histórica, interiores, dirección, descripción de la imagen estén llenos. 2.2 El sistema verifica que esta Residencia no exista. 2.3 La residencia se almacena en el sistema. 2.4 Se muestra un mensaje informándosele al administrador que ya ha sido efectuado el registro de la residencia.
Flujo alternativo	
	2.3 Se emite un mensaje para que llene los campos obligatorios. 2.4 Si la residencia existe se muestra un mensaje informativo y finaliza el caso de uso.
Sección "Listar Residencias"	
1. El administrador del sistema selecciona el idioma en que desea listar los datos de la residencia.	1.1 El sistema muestra un listado con los datos número de ip, nivel, zona, número de cada residencia que se encuentra en el sistema.
Sección "Editar Residencia "	
Acciones del Actor	**Respuesta del Sistema**
1.El administrador del sistema selecciona el idioma en que desea modificar la residencia.	
2.El administrador del sistema selecciona la residencia a modificar.	2.1 El sistema brinda la posibilidad de modificar los datos.
3.El administrador del sistema realiza las actualizaciones deseadas.	3.1Se verifica que los campos obligatorios estén llenos. 3.2 Se actualiza la Información.
Flujo alternativo	
Línea 3.2	
	3.2 Se emite un mensaje para que llene los campos obligatorios.
Sección "Eliminar Residencia "	
Acciones del Actor	**Respuesta del Sistema**
1. El administrador del sistema selecciona la residencia a eliminar.	1.1 Es sistema pide confirmación. 1.2 El sistema elimina la residencia.
Puntos de Extensión	

Las descripciones correspondientes a los casos de uso:

- Gestionar Ingrediente.
- Gestionar Insumo.
- Gestionar Obra de Arte.
- Gestionar Sitio Web.
- Gestionar Persona.
- Gestionar Imagen de Galería.

Se omiten porque son similares a la descripción del caso de uso Gestionar Residencia, pues presentan el mismo flujo de actividades.

Conclusiones

En el capítulo se definieron los requerimientos funcionales y no funcionales que debía cumplir el sistema, se definieron los actores del sistema, los casos de uso del sistema y la relación entre ambos, se describieron paso a paso las acciones de los actores y las respuestas que el sistema ofrece ante cada acción.

Capítulo III

Análisis y Diseño

En este capítulo se confeccionarán los diagramas de clases de análisis y de interacción correspondientes a cada caso de uso del sistema. Se definirá el diseño de la Intranet a través de un diagrama de clases del diseño para cada caso de uso del sistema y se explicarán las definiciones del diseño que se aplique. Se describirán las clases del diseño especificando sus atributos y métodos y se conformará el diagrama de entidad-relación de las clases persistentes.

3.1 Diagramas de Clases del Análisis.

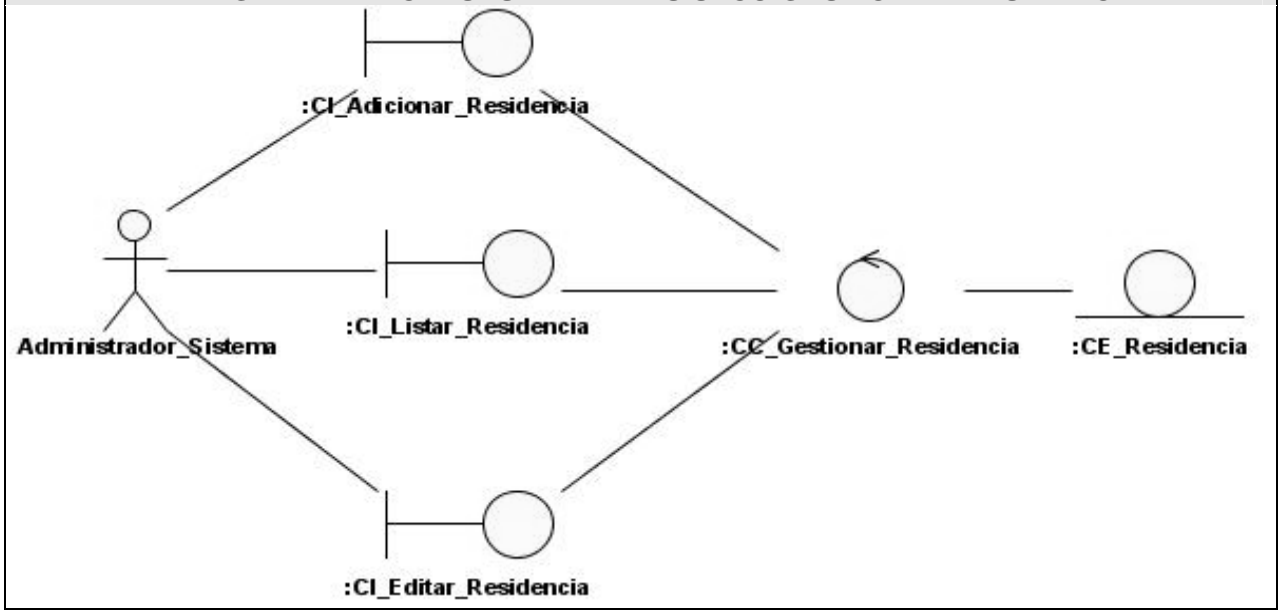
En la construcción del modelo de análisis se identifican las clases que describen la realización de los casos de uso y las relaciones entre ellas. Con esta información se construye el diagrama de clases del análisis, que se descompone para agrupar las clases en paquetes. Estas clases son: Clase Interfaz, Clase Controladora y Clase Entidad.

Clase Interfaz: Modela la interacción entre el sistema y sus actores.

Clase Controladora: Coordinan el flujo de trabajo de las clases, encapsulando el comportamiento de un caso de uso.

Clase Entidad: Modelan la información del sistema.

DIAGRAMA DE CLASES DE ANÁLISIS: CU GESTIONAR RESIDENCIA



- Los Diagramas de Clases del Análisis correspondientes a los casos de uso:

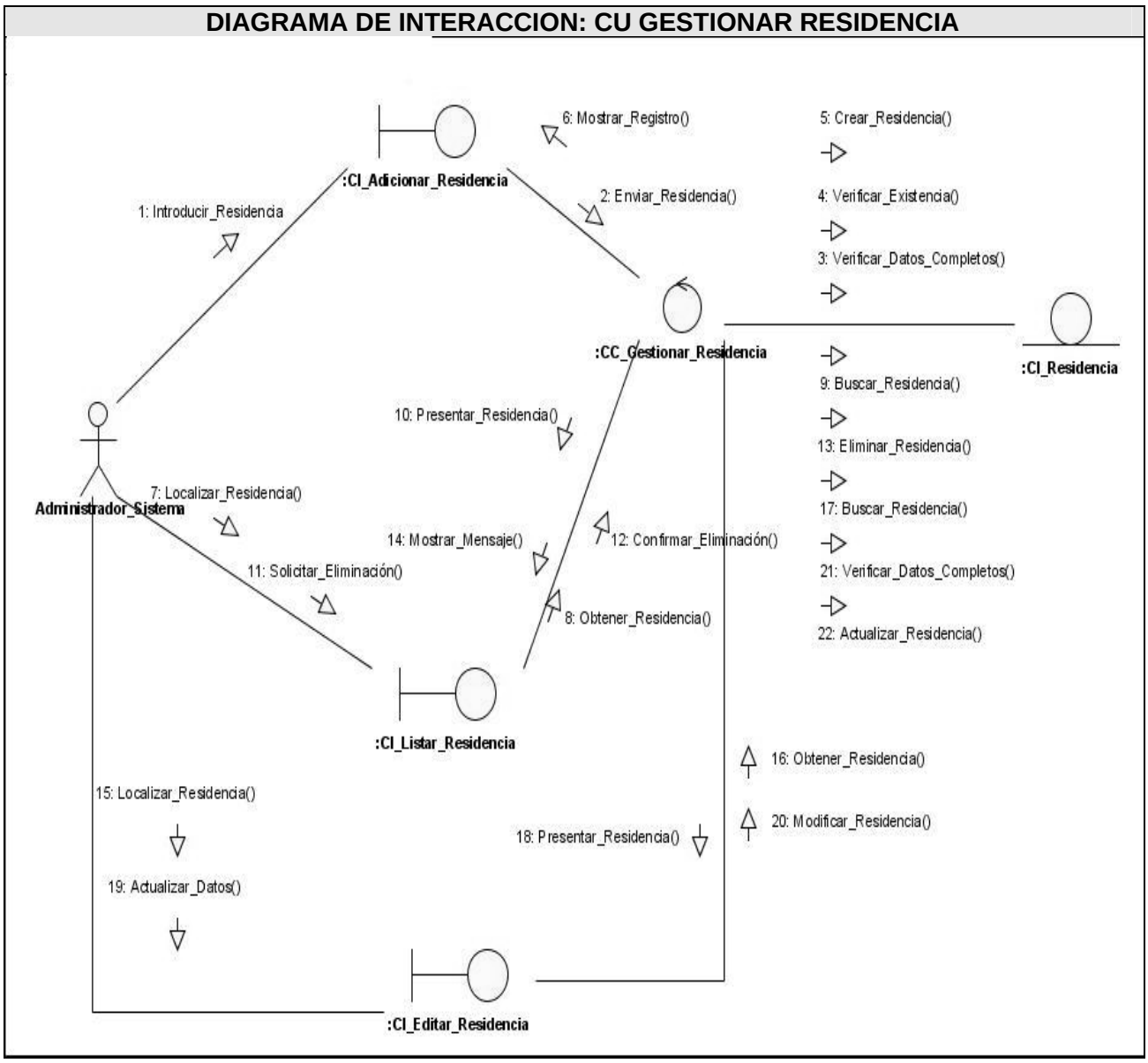
  - Gestionar Usuario
  - Gestionar Obra de Arte
  - Gestionar Sitio Web
  - Gestionar Imagen de Galería
  - Gestionar Persona
  - Gestionar Insumo
  - Gestionar Ingrediente
  - Gestionar Rol

Se omiten porque son similares al Diagrama de Clases de Análisis del caso de uso Gestionar Residencia, es decir, todos estos casos de uso poseen una clase interfaz: Adicionar, Listar y Editar, de la misma manera presentan una clase controladora y una clase entidad correspondiente a cada caso de uso.

Para consultar el resto de los Diagramas de Clases del Análisis remítase a los anexos en la versión digital contenida en el CD adjunto.

3.2 Diagramas de Interacción (Diagramas de Colaboración).

El diagrama de colaboración destaca la organización de los objetos que participan en una interacción, representando los enlaces y mensajes entre ellos.



- Los Diagramas de Colaboración correspondientes a los casos de uso:

  - Gestionar Usuario
  - Gestionar Obra de Arte
  - Gestionar Sitio Web
  - Gestionar Imagen de Galería
  - Gestionar Persona
  - Gestionar Insumo
  - Gestionar Ingrediente

Se omiten porque son similares al Diagrama de Colaboración del caso de uso Gestionar Residencia, es decir, todos estos casos de uso poseen de manera muy similar el flujo de mensajería entre las clases interfaz, controladora y entidades correspondientes a cada caso de uso.

Para consultar el resto de los Diagramas de Colaboración remítase a los anexos en la versión digital contenida en el CD adjunto.

3.3 Diagrama de Clases del Diseño

Los diagramas de clases muestran el diseño del sistema desde un punto de vista estático, a través de una colección de elementos declarativos, como clases, colaboraciones y sus relaciones.

Al ser la Intranet una aplicación Web, la misma se modelará con estereotipos definidos para este tipo de aplicaciones, como son los estereotipos Web, los que proporcionan una mayor comprensión de las funcionalidades del sistema y el poder distinguir sus atributos, operaciones y relaciones.

A continuación se brinda una explicación de qué representa cada estereotipo en el diseño:

 sp_<NombreClase Servidora>	<<Server Page>>: Representa la clase que tiene código que se ejecuta en el servidor, la cual se encarga de construir (build) o generar el resultado HTML y/o realizar peticiones a la capa inferior.
 cp_<NombreClase Cliente>	<<Client Page>>: Es una página Web con formato HTML/XHTML. Son interpretadas por el navegador. Sus atributos son las variables declaradas dentro del script que son accesibles para cualquier función dentro de la página. Cada página cliente es construida por una sola página de servidor.

 <Nombre>	<<FormHTML>>: Es una colección de elementos de entrada que están contenidos en la página cliente. Sus atributos son los elementos de entrada del formulario (input boxes, text areas, radio buttons, check boxes, hidden fields, entre otros). No tienen operaciones, el método para el paso de los parámetros es $_POST y se comunican con las páginas servidores mediante submit.

Relaciones utilizadas entre clases:

<<build>>: Representa la relación existente entre las páginas cliente, que de forma general expresa cómo las páginas que se encuentran en el servidor construyen las páginas en el cliente. Es una relación direccional, donde una página servidor construye una o más páginas cliente.

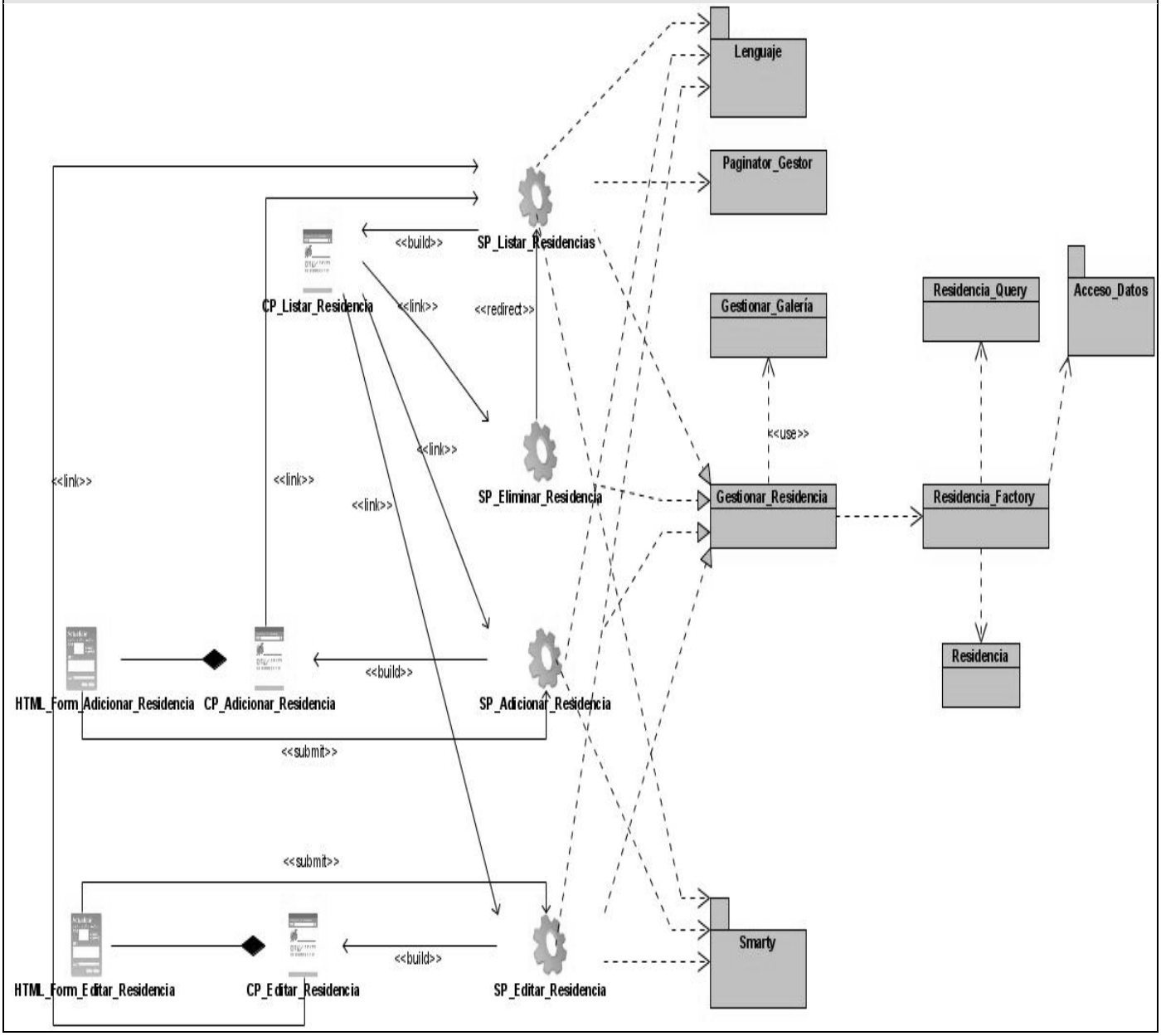
<<redirect>>: Una página servidora puede redireccionar el procesamiento a otra página, es decir, enviar información para que la otra ejecute la acción.

<<use>>: Representa herencia o instanciación entre clases.

<<Link>> Expresa las asociaciones más comunes entre las páginas, en este caso la del hipervínculo; esta asociación siempre se origina desde una página cliente y apunta hacia otra página cliente o una página de servidor.

<<Submit>> Es la relación que se crea siempre entre una página servidor y un formulario, a través de esta relación el formulario manda los valores de sus campos al servidor, para ser procesados por la página servidor.

DIAGRAMA DE CLASES DEL DISEÑO: CU GESTIONAR RESIDENCIA



- Los Diagramas de Diseño correspondientes a los casos de uso:

  - Gestionar Usuario
  - Gestionar Obra de Arte
  - Gestionar Sitio Web
  - Gestionar Imagen de Galería
  - Gestionar Persona
  - Gestionar Insumo
  - Gestionar Ingrediente

Se omiten porque son similares al Diagrama de Diseño del caso de uso Gestionar Residencia.

DIAGRAMA DE CLASES DEL DISENO: CU AUTENTICAR USUARIO

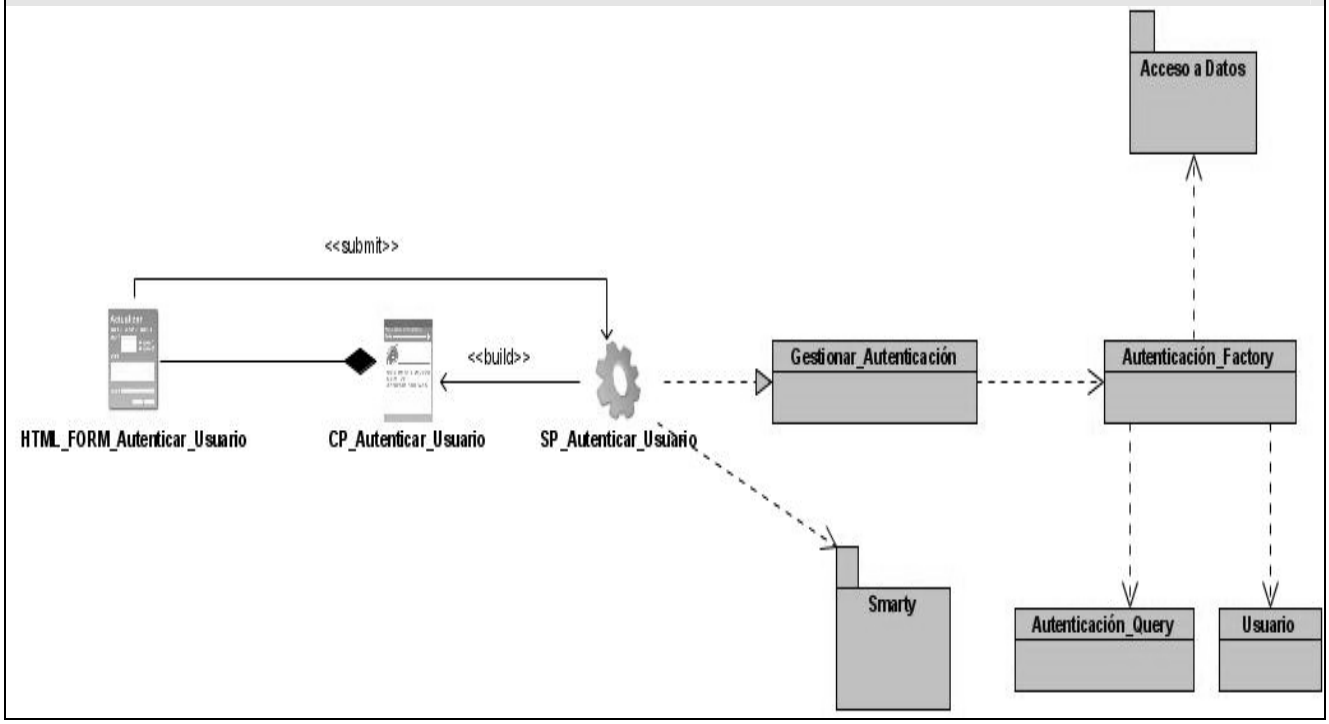
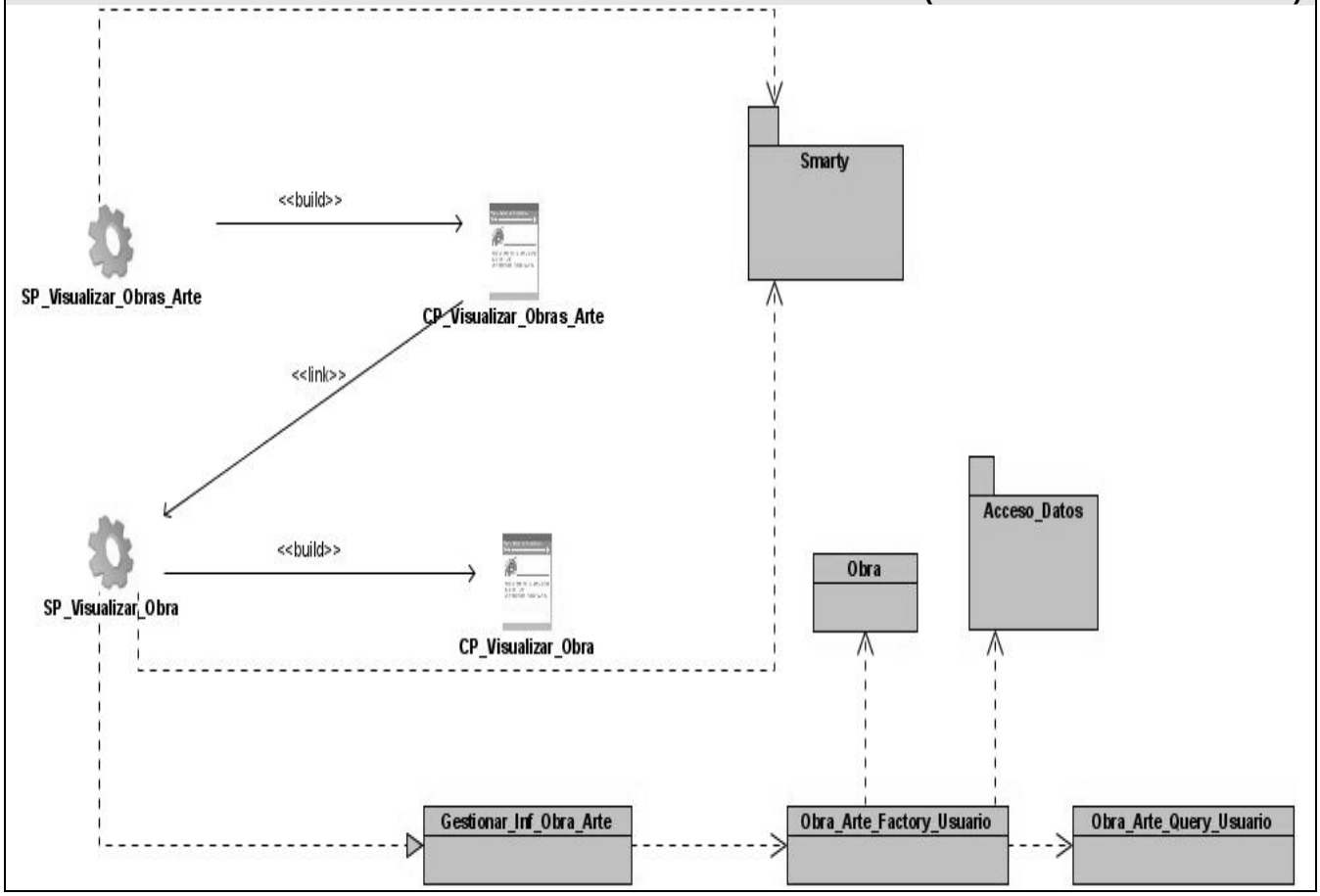


DIAGRAMA DE CLASES DEL DISENO: CU VISUALIZAR INFORMACION (ESCENARIO OBRA DE ARTE)



Para consultar el resto de los Diagramas de Diseño remítase a los anexos en la versión digital contenida en el CD adjunto.

3.4 Definiciones del Diseño.

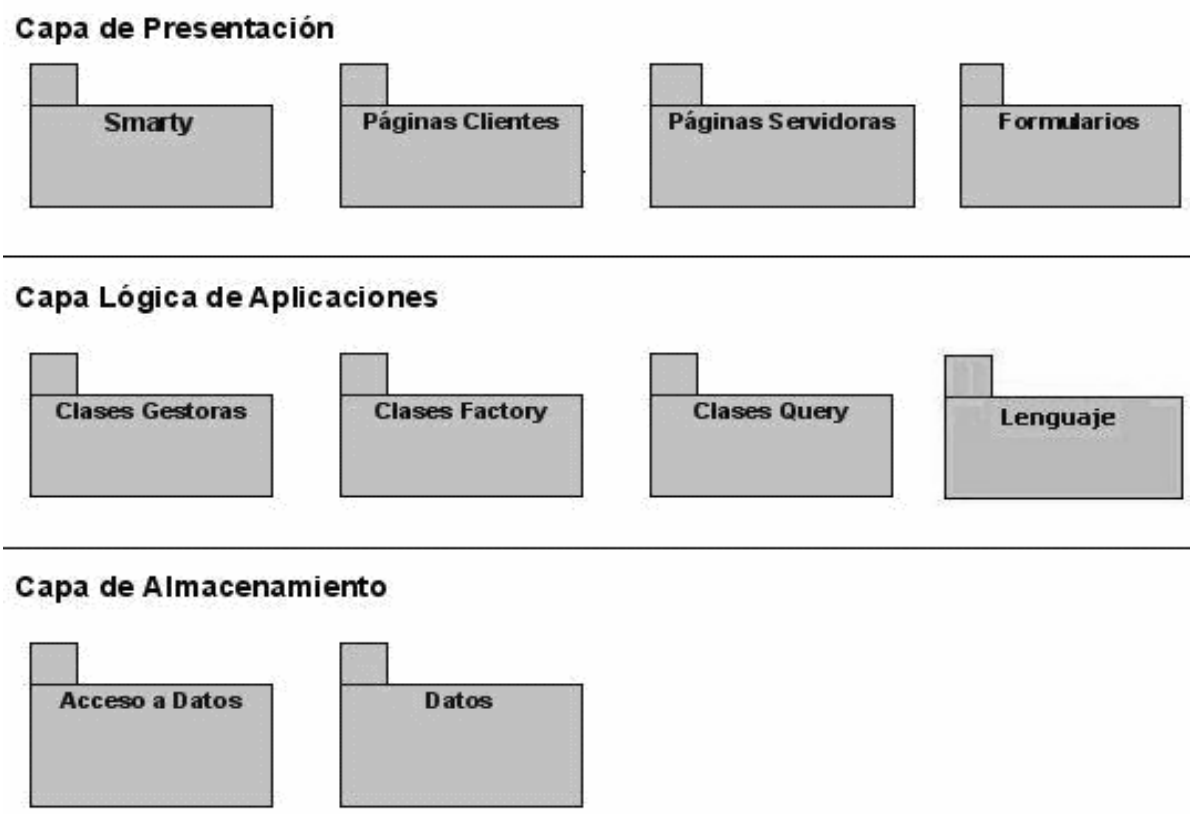
El diseño de la Intranet para las Residencias de Protocolo está basado en la arquitectura clásica de tres capas, este es un patrón de diseño que abarca una interfaz para el usuario y el almacenamiento persistente de los datos. La arquitectura está dividida en una capa de presentación, formada por ventanas, formularios, la lógica de aplicaciones donde se encuentran las tareas y reglas que rigen el proceso y la capa de almacenamiento donde están los mecanismos de almacenamiento persistente.

La calidad tan especial de la arquitectura de tres capas consiste en aislar la lógica de la aplicación y convertirla en una capa intermedia bien definida y lógica del software. En la capa de presentación se realiza relativamente poco procesamiento de la aplicación; las ventanas envían a la capa intermedia peticiones de trabajo y este se comunica con la capa de almacenamiento del extremo posterior.

En el diseño de la Intranet las clases que van a formar parte de la Capa de Presentación son las clases del paquete smarty que es una librería de clases para separar el código PHP del código HTML, las páginas servidoras, las páginas clientes y los formularios. Por otra parte las clases que conforman la Capa de Lógica de Aplicaciones serán las clases gestoras que contiene los métodos para realizar las acciones sobre los datos, las clases factory que contiene los métodos que ejecutan las consultas a la base de datos, la clase query que guarda las consultas a la base de datos y las clases del paquete lenguaje que contienen los métodos para configurar el idioma de la información .

Mientras que la Capa de Almacenamiento agrupará todo el conjunto de clases que guardan los datos y el paquete de acceso a datos que contiene las clases que permiten la conexión con la base de datos.

Las tres capas se pueden separar según el esquema:



El diseño propuesto cumple con el patrón de alta cohesión, pues las clases poseen responsabilidades estrechamente relacionada y colaboran entres ellas en un área determinada para llevar a cabo las tareas, por ejemplo la clase controladora “Gestionar Residencia” es la que coordina y encapsula todo el comportamiento solamente del caso de uso Gestionar Residencia.

El diseño también cumple con el patrón de bajo acoplamiento, pues el funcionamiento de una clase, no depende del funcionamiento de muchas otras. Volviendo a utilizar como ejemplo la clase “Gestionar Residencia”, se observa que esta solo depende de la clase “Residencia Factory” ambas ubicadas dentro de una misma área de acción.

3.5 Descripciones de las Clases del Diseño.

Nombre: Gestionar_Encuesta_Opinión.	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getPregunta($lang)
Descripción:	Devuelve la pregunta de una encuesta en el lenguaje que recibe como parámetro de entrada.

Nombre:	getRespuestasByEnc($lang, $idenc)
Descripción:	Devuelve las respuestas de una encuesta, en el lenguaje que recibe como parámetro de entrada.
Nombre:	addVoto($ipr, $fech, $ide, $idr)
Descripción:	Crea una nueva instancia en la tabla de la base de datos que guarda los datos de los votos con los valores que recibe como parámetro de entrada.
Nombre:	existeVoto($ipr, $fech, $idenc)
Descripción:	Devuelve verdadero si existe en la base de datos un voto con los valores que recibe como parámetros de entrada y falso si no existe.
Nombre:	addOpinion($ipr,$ids,$idc,$nom, $opn, $fech)
Descripción:	Crea una nueva instancia en la tabla de la base de datos que guarda los datos de las opiniones con los valores que recibe como parámetro de entrada.
Nombre:	getVotosRespuesta($idR, $idE)
Descripción:	Devuelve la cantidad de votos que tiene una respueta.
Nombre:	getVotosEncuesta($idE)
	Devuelve la cantidad de votos que tiene una encuesta.
Nombre:	getCalificaciones($lang)
Descripción:	Devuelve las calificaciones para un servicio en el lenguaje que recibe como parámetro de entrada.

Nombre: Encuesta_Opinión_Factory	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getPregunta($lang)
Descripción:	Ejecuta la consulta a la base de datos que devuelve las preguntas de las encuestas activas, en el lenguaje que recibe como parámetro de entrada.
Nombre:	getRespuestasByEnc($lang, $idenc)
Descripción:	Ejecuta la consulta a la base de datos que devuelve las respuestas de una encuesta, en el lenguaje que recibe como parámetro de entrada.
Nombre:	addVoto($ipr, $fech, $ide, $idr)
Descripción:	Ejecuta la consulta a la base de datos que adiciona un voto a la misma.
Nombre:	existeVoto($ipr, $fech, $idenc)
Descripción:	Ejecuta la consulta a la base de datos que devuelve verdadero si existe un voto con los valores que recibe como parámetros de entrada y falso si no existe.
Nombre:	addOpinion($ipr,$ids,$idc,$nom, $opn, $fech)
Descripción:	Ejecuta la consulta a la base de datos que adiciona una nueva opinión a la misma.
Nombre:	getVotosRespuesta($idR, $idE)
Descripción:	Ejecuta la consulta a la base de datos que devuelve la cantidad de votos que tiene una respuesta.
Nombre:	getVotosEncuesta($idE)
Descripción:	Ejecuta la consulta a la base de datos que devuelve la cantidad de votos que tiene una encuesta.
Nombre:	getCalificaciones($lang)
Descripción:	Ejecuta la consulta a la base de datos que devuelve las calificaciones para un servicio en el lenguaje que recibe como parámetro de entrada.

Nombre: Gestionar_Inf_Residencia	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	listarResidenciaZona($zona, $lang)
Descripción:	Devuelve una lista con todos los nombres de las residencias de una zona, en el lenguaje que recibe como parámetro de entrada.
Nombre:	getHistoriaDireccion($ip, $lang)
Descripción:	Devuelve la historia de una residencia, en el lenguaje que recibe como parámetro de entrada.
Nombre:	getInteriores($ip, $lang)
Descripción:	Devuelve la información del interior de una residencia, en el lenguaje que recibe como parámetro de entrada.
Nombre:	getNombreImagen($ip, $lang)
Descripción:	Devuelve el nombre de la imagen de una residencia, en el lenguaje que recibe como parámetro de entrada.
Nombre:	getResidenciaId($zona, $num, $lang)
Descripción:	Devuelve el nombre de una residencia cuya zona, número e idioma recibe como parámetro de entrada.
Nombre:	getPersona($ipR, $lang)
Descripción:	Devuelve la información del personal de una residencia, en el lenguaje que recibe como parámetro de entrada.
Nombre:	getResidenceNameByIp(**$**ipR)
Descripción:	Devuelve el nombre de una residencia cuyo dirección ip recibe como parámetro de entrada.

Nombre: Residencia_Factory_Usuario	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	listarResidenciaZona($zona, $lang)
Descripción:	Ejecuta la consulta a la base de datos que devuelve una lista con todos los nombres de las residencias de una zona, en el lenguaje que recibe como parámetro de entrada.
Nombre:	getHistoriaDireccion($ip, $lang)
Descripción:	Ejecuta la consulta a la base de datos que devuelve la historia de una residencia, en el lenguaje que recibe como parámetro de entrada.
Nombre:	getInteriores($ip, $lang)
Descripción:	Ejecuta la consulta a la base de datos que devuelve la información del interior de una residencia, en el lenguaje que recibe como parámetro de entrada.
Nombre:	getNombreImagen($ip, $lang)
Descripción:	Ejecuta la consulta a la base de datos que devuelve el nombre de la imagen de una residencia, en el lenguaje que recibe como parámetro de entrada.

Nombre:	getResidenciaById($zona, $num, $lang)
Descripción:	Ejecuta la consulta a la base de datos que devuelve el nombre de una residencia cuya zona, número e idioma recibe como parámetro de entrada.
Nombre:	getPersona($ipR, $lang)
Descripción:	Ejecuta la consulta a la base de datos que devuelve la información del personal de una residencia, en el lenguaje que recibe como parámetro de entrada.
Nombre:	getResidenceNameByIp($ipR)
Descripción:	Ejecuta la consulta a la base de datos que devuelve el nombre de una residencia cuya dirección ip recibe como parámetro de entrada.

Nombre: Gestionar_Inf_Obra_Arte	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getIdObraActiva()
Descripción:	Devuelve el identificador de la obra.
Nombre:	getObras($lang)
Descripción:	Devuelve los datos de las obras que están en una residencia, en el lenguaje que recibe como parámetro de entrada.
Nombre:	getObrasInfo($id, $lang)
Descripción:	Devuelve los datos de una obra en el lenguaje que recibe como parámetro de entrada.

Nombre: Obra_Arte_Factory_Usuario	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getIdObraActiva()
Descripción:	Ejecuta la consulta de la base de datos que devuelve el identificador de la obra.
Nombre:	getObras($ip, $lang)
Descripción:	Ejecuta la consulta de la base de datos que devuelve los datos de las obras que están en una residencia con esa dirección de ip y en el lenguaje que recibe como parámetro de entrada.
Nombre:	getObrasInfo($id, $lang)
Descripción:	Ejecuta la consulta de la base de datos que devuelve los datos de una obra en el lenguaje que recibe como parámetro de entrada.

Nombre: Gestionar_Inf_Gastronomía	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getCocteles($lang)
Descripción:	Devuelve los datos de los cocteles en el lenguaje que recibe como parámetro de entrada.
Nombre:	getCategoriasCoct($lang)
Descripción:	Devuelve las categorías de los cocteles en el lenguaje que recibe como parámetro de entrada.
Nombre:	getBebidas($lang)
Descripción:	Devuelve los datos de las bebidas en el lenguaje que recibe como parámetro de entrada.
Nombre:	getCategoriasBeb($lang)
Descripción:	Devuelve las categorías de las bebidas en el lenguaje que recibe como parámetro de entrada.
Nombre:	getVinos($lang)
Descripción:	Devuelve los datos de los vinos en el lenguaje que recibe como parámetro de entrada.
Nombre:	getNivelRes($ip)
Descripción:	Devuelve el nivel de una residencia.
Nombre:	getCategoriasVn($lang)
Descripción:	Devuelve las categorías de los vinos en el lenguaje que recibe como parámetro de entrada.
Nombre:	getCategoriasMn($tipo, $lang)
Descripción:	Devuelve las categorías del los menús en el lenguaje que recibe como parámetro de entrada.
Nombre:	getIngredienteMenu($lang, $idC, $idS, $tipo)
Descripción	Devuelve los ingredientes de un menú.
Nombre:	getDiaSemanaMenu($lang)
Descripción:	Devuelve el día de la semana de un menú en el lenguaje que recibe como parámetro de entrada.
Nombre:	getMenuAlmuerzo($lang)
Descripción:	Devuelve los datos de los menús de almuerzo en el lenguaje que recibe como parámetro de entrada.
Nombre:	getMenuCena($lang)
Descripción:	Devuelve los datos de los menús de la cena en el lenguaje que recibe como parámetro de entrada.

Nombre: Gastronomía_Factory	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getInsumo($lang)
Descripción:	Ejecuta la consulta que devuelve los datos del insumo en el lenguaje que recibe como parámetro de entrada.

Nombre:	getCategoriasInsumo($lang)
Descripción:	Ejecuta la consulta que devuelve las categorías del insumo en el lenguaje que recibe como parámetro de entrada.
Nombre:	getInsumo($lang)
Descripción:	Ejecuta la consulta que devuelve los datos del insumo en el lenguaje que recibe como parámetro de entrada.
Nombre:	getInsumo($lang)
Descripción:	Ejecuta la consulta que devuelve los datos de los insumos en el lenguaje que recibe como parámetro de entrada.
Nombre:	getNivelRes($ip)
Descripción:	Devuelve el nivel de una residencia.
Nombre:	getCategoriasMn($tipo, $lang)
Descripción:	Ejecuta la consulta que devuelve las categorías del los menús en el lenguaje que recibe como parámetro de entrada.
Nombre:	getIngredienteMenu($lang, $idC, $idS, $tipo)
Descripción	Ejecuta la consulta que devuelve los ingredientes de un menú.
Nombre:	getDiaSemanaMenu($lang)
Descripción:	Ejecuta la consulta que devuelve el día de la semana de un menú en el lenguaje que recibe como parámetro de entrada.

Nombre: Gestionar_Inf_Solicitud_Lavanderia	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	SolicitarServ($array)
Descripción:	Crea una nueva instancia en la tabla de la base de datos que guarda los datos de las solicitudes de lavandería con los valores que recibe como parámetro de entrada.

Nombre: Lavandería_Factory_Usuario	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	solicitarServicio($tipo_serv, $nom, $fecha, $hora, $ipR, $hab)
Descripción:	Ejecuta la consulta a la base de datos que crea una nueva instancia en la tabla de la base de datos que guarda los datos de las solicitudes de lavandería con los valores que recibe como parámetro de entrada.

Nombre: Gestionar_Inf_Galería	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()

Descripción:	Construye la clase.
Nombre:	getImagen($lang, $idcat)
Descripción:	Devuelve los datos de las imágenes de una categoría en el lenguaje que recibe como parámetro de entrada.
Nombre:	getCategorias($lang)
Descripción:	Devuelve las categorías de las imágenes en el lenguaje que recibe como parámetro de entrada.
Nombre:	getImagenById($id, $lang)
Descripción:	Devuelve los datos de una imagen en el lenguaje que recibe como parámetro de entrada.
Nombre:	getImagePathById($id)
Descripción:	Devuelve la dirección donde se encuentra guardada la imagen cuyo identificador recibe como parámetro de entrada.

Nombre: Galería_Factory_Usuario	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getImagen($lang, $idcat)
Descripción:	Ejecuta la consulta que devuelve los datos de las imágenes de una categoría en el lenguaje que recibe como parámetro de entrada.
Nombre:	getCategorias($lang)
Descripción:	Ejecuta la consulta que devuelve las categorías de las imágenes en el lenguaje que recibe como parámetro de entrada.
Nombre:	getImagenById($id, $lang)
Descripción:	Ejecuta la consulta que devuelve los datos de una imagen en el lenguaje que recibe como parámetro de entrada.
Nombre:	getImagePathById($id)
Descripción:	Ejecuta la consulta que devuelve la dirección de la carpeta servidora donde se encuentra guardada la imagen cuyo identificador recibe como parámetro de entrada.

Nombre: Gestionar_Sitio_Web	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getSitio($lang)
Descripción:	Devuelve los datos de los sitios web, en el lenguaje que recibe como parámetro de entrada.
Nombre:	getCategorias($lang)
Descripción:	Devuelve las categorías de los sitios web, en el lenguaje que recibe como parámetro de entrada.
Nombre:	addSitio($array)
Descripción:	Crea una nueva instancia en la tabla de la base de datos que guarda los datos de los sitios web con los valores que recibe como parámetro de

	entrada.
Nombre:	deleteSitio($id)
Descripción:	Elimina los datos del sitio web cuyo identificador recibe como parámetro de entrada.
Nombre:	getSitioById($id, $lang)
Descripción:	Devuelve los datos de un sitio web, en el lenguaje que recibe como parámetro de entrada.
Nombre:	editSitio($array)
Descripción:	Modifica los datos de un sitio web sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	existeSitioLeng($idS, $idLeng)
Descripción:	Devuelve verdadero si existe en la base de datos un sitio web con los valores que recibe como parámetros de entrada y falso si no existe.

Nombre: Sitio_Web_Factory	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getSitio($lang)
Descripción:	Ejecuta la consulta que devuelve los datos de los sitios web, en el lenguaje que recibe como parámetro de entrada.
Nombre:	getCategorias()
Descripción:	Ejecuta la consulta que devuelve las categorías de los sitios web.
Nombre:	addSitio($idCat, $link)
Descripción:	Ejecuta la consulta que crea una nueva instancia en la tabla de la base de datos que guarda los datos de los sitios web con los valores que recibe como parámetro de entrada.
Nombre:	getLastInsert()
Descripción:	Ejecuta la consulta que devuelve el identificador de la última instancia de un sitio web.
Nombre:	addSitioLenguage($idLeng, $idSitio, $titulo, $desc)
Descripción:	Ejecuta la consulta que guarda los datos del sitio web en el lenguaje que recibe como parámetro de entrada.
Nombre:	deleteSitio($id)
Descripción:	Ejecuta la consulta que elimina los datos del sitio web cuyo identificador recibe como parámetro de entrada.
Nombre:	getSitioById($id, $lang)
Descripción:	Ejecuta la consulta que devuelve verdadero si existe en la base de datos una residencia con el valor de IP que recibe como parámetros de entrada y falso si no existe.
Nombre:	editResidencia($idLeng, $ipR, $zona, $nivel, $numero, $historia, $interiores, $direccion, $ipH, $nombre)
Descripción:	Ejecuta la consulta que modifica los datos de una residencia sustituyéndolos por los datos que recibe como parámetros de entrada.

Nombre: Gestionar_Usuario	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getUsuarios()
Descripción:	Devuelve los datos de los usuarios.
Nombre:	addUsuario($array)
Descripción:	Crea una nueva instancia en la tabla de la base de datos que guarda los datos de los usuarios con los valores que recibe como parámetro de entrada.
Nombre:	delUsuario($id))
Descripción:	Elimina los datos de un usuario cuyo identificador recibe como parámetro de entrada.
Nombre:	getUsurioById($id)
Descripción:	Devuelve los datos de un usuario, en el lenguaje que recibe como parámetro de entrada.
Nombre:	editUsuario($array)
Descripción:	Modifica los datos de un usuario sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	existeUsuario($user)
Descripción:	Devuelve verdadero si existe en la base de datos un usuario con el dato que recibe como parámetros de entrada y falso si no existe.
Nombre:	editPermisos($array)
Descripción:	Modifica los datos de los permisos de los usuarios sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	getModulos($idU)
Descripción:	Devuelve el módulo correspondiente a los usuarios cuyo identificador recibe como parámetro de entrada.
Nombre:	getDptos()
Descripción:	Devuelve el departamento al que pertenece los usuarios.

Nombre: Usuario_Factory	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getUsuarios()
Descripción:	Ejecuta la consulta que devuelve los datos de los usuarios.
Nombre:	addUsuario($array)
Descripción:	Ejecuta la consulta que crea una nueva instancia en la tabla de la base de datos que guarda los datos de los usuarios con los valores que recibe como parámetro de entrada.
Nombre:	delUsuario($id))
Descripción:	Ejecuta la consulta que elimina los datos de un usuario cuyo identificador recibe como parámetro de entrada.
Nombre:	getUsurioById($id)

Descripción:	Ejecuta la consulta que devuelve los datos de un usuario, en el lenguaje que recibe como parámetro de entrada.
Nombre:	editUsuario($array)
Descripción:	Ejecuta la consulta que modifica los datos de un usuario sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	existeUsuario($user)
Descripción:	Ejecuta la consulta que devuelve verdadero si existe en la base de datos un usuario con el dato que recibe como parámetros de entrada y falso si no existe.
Nombre:	editPermisos($array)
Descripción:	Ejecuta la consulta que modifica los datos de los permisos de los usuarios sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	getModulos($idU)
Descripción:	Ejecuta la consulta que devuelve el módulo correspondiente a los usuarios cuyo identificador recibe como parámetro de entrada.
Nombre:	getModulosAcceso($idU)
Descripción:	Ejecuta la consulta que da el acceso a los módulos en dependencia del usuario que recibe como parámetro de entrada.
Nombre:	getDptos()
Descripción:	Ejecuta la consulta que devuelve el departamento al que pertenecen los usuarios.

Nombre: Gestionar_Encuesta	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getEncuestas($lang)
Descripción:	Devuelve los datos de las encuestas en el lenguaje que recibe como parámetro de entrada.
Nombre:	addEncuesta($array)
Descripción:	Crea una nueva instacia en la tabla de la base de datos que guarda los datos de las encuestas con los valores que recibe como parámetro de entrada.
Nombre:	delEncuesta($idE)
Descripción:	Elimina los datos de una encuesta cuyo identificador recibe como parámetro de entrada.
Nombre:	getEncuestaById($id, $lang)
Descripción:	Devuelve los datos de una encuesta, en el lenguaje que recibe como parámetro de entrada.
Nombre:	editEncuesta($array)
Descripción:	Modifica los datos de una encuesta sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	activarEnc($ide)
Descripción:	Hace disponible el formulario de la encuesta que recibe como parámetro de entrada.
Nombre:	desactivarEnc($ide)
Descripción:	Quita la disponibilidad del formulario de la encuesta que recibe como parámetro de entrada.

Nombre:	resetEncuesta($idE)
Descripción:	Coloca en 0% los votos de las respuestas de la encuesta.
Nombre:	addRespuesta($array)
Descripción:	Crea una nueva instancia en la tabla de la base de datos que guarda los datos de las respuestas con los valores que recibe como parámetro de entrada.
Nombre:	getRespuestas($lang)
Descripción:	Devuelve los datos de una respuesta en el lenguaje que recibe como parámetro de entrada.
Nombre:	getRespuestaByEnc($idenc, $lang)
Descripción:	Devuelve los datos de una respuesta, en el lenguaje que recibe como parámetro de entrada.
Nombre:	editRespuesta($array)
Descripción:	Modifica los datos de una respuesta sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	delRespuestaId($idR)
Descripción:	Elimina los datos de una respuesta cuyo identificador recibe como parámetro de entrada.

Nombre: Encuesta_Factory	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getEncuestas($lang)
Descripción:	Ejecuta la consulta que devuelve los datos las encuestas en el lenguaje que recibe como parámetro de entrada.
Nombre:	addEncuesta($fecha_pub, $fecha_exp)
Descripción:	Ejecuta la consulta que crea una nueva instancia en la tabla de la base de datos que guarda los datos de las encuestas con los valores que recibe como parámetro de entrada.
Nombre:	addEncuestaPregunta($idEn, $idleng, $tit)
Descripción:	Ejecuta la consulta que crea una nueva instancia en la tabla de la base de datos que guarda los datos de las encuestas con los valores que recibe como parámetro de entrada.
Nombre:	delEncuesta($idE)
Descripción:	Ejecuta la consulta que elimina los datos de una encuesta cuyo identificador recibe como parámetro de entrada.
Nombre:	getEncuestaById($id, $lang)
Descripción:	Ejecuta la consulta que devuelve los datos de una encuesta, en el lenguaje que recibe como parámetro de entrada.
Nombre:	editEncuesta($idEn, $fecha_pub, $fecha_ext)
Descripción:	Ejecuta la consulta que modifica los datos de una encuesta sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	getVotosEncuestas($idE)
Descripción:	Ejecuta la consulta que devuelve los votos de las encuestas.
Nombre:	editEncuestaPregunta($idE, $idL, $tit)
Descripción:	Ejecuta la consulta que modifica los datos de una encuesta

	sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	getLastEncuesta()
Descripción:	Ejecuta la consulta que devuelve los datos las encuestas de forma ordenada.
Nombre:	activarEnc($ide)
Descripción:	Ejecuta la consulta que pone disponible el formulario de la encuesta que recibe como parámetro de entrada.
Nombre:	desactivarEnc($ide)
Descripción:	Ejecuta la consulta que quita la disponibilidad del formulario de la encuesta que recibe como parámetro de entrada.
Nombre:	resetEncuesta($idE)
Descripción:	Ejecuta la consulta que coloca en 0% los votos de las respuestas de la encuesta.
Nombre:	addRespuesta($idEn)
Descripción:	Ejecuta la consulta que crea una nueva instancia en la tabla de la base de datos que guarda los datos de las respuestas con los valores que recibe como parámetro de entrada.
Nombre:	addRespuestaLeng($idR, $idL, $tit)
Descripción:	Ejecuta la consulta que crea una nueva instancia en la tabla de la base de datos que guarda los datos de las respuestas con los valores que recibe como parámetro de entrada.
Nombre:	getRespuestas($lang)
Descripción:	Ejecuta la consulta que devuelve los datos de una respuesta en el lenguaje que recibe como parámetro de entrada.
Nombre:	getLastRespuesta()
Descripción:	Ejecuta la consulta que devuelve los datos de las respuestas de forma ordenada.
Nombre:	getVotosRespuesta($idR)
Descripción:	Ejecuta la consulta que devuelve los votos de las respuestas.
Nombre:	editRespuesta($idR, $idL, $tit)
Descripción:	Ejecuta la consulta que modifica los datos de una respuesta sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	delRespuestaId($idR)
Descripción:	Ejecuta la consulta que elimina los datos de una respuesta cuyo identificador recibe como parámetro de entrada.
Nombre:	delRespuesta($idRes, $idEn)
Descripción:	Ejecuta la consulta que elimina los datos de una respuesta cuyo identificador recibe como parámetro de entrada.

Nombre: Gestionar_Rol	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getRol()
Descripción:	Devuelve los datos de los roles.
Nombre:	addRol($array)
Descripción:	Crea una nueva instancia en la tabla de la base de datos que guarda los datos de los roles con los valores que recibe como parámetro de entrada.

Nombre:	delRol($id))
Descripción:	Elimina los datos de un rol cuyo identificador recibe como parámetro de entrada.
Nombre:	getRolById($id)
Descripción:	Devuelve los datos de un rol, en el lenguaje que recibe como parámetro de entrada.
Nombre:	editRol($array)
Descripción:	Modifica los datos de un rol sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	existeRol($idrol)
Descripción:	Devuelve verdadero si existe en la base de datos un usuario con el mismo identificador que recibe como parámetros de entrada y falso si no existe.

Nombre: Rol_Factory	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getUsuarios()
Descripción:	Ejecuta la consulta que devuelve los datos de los roles.
Nombre:	addRol($array)
Descripción:	Ejecuta la consulta que crea una nueva instancia en la tabla de la base de datos que guarda los datos de los roles con los valores que recibe como parámetro de entrada.
Nombre:	delRol($id))
Descripción:	Ejecuta la consulta que elimina los datos de un usuario cuyo identificador recibe como parámetro de entrada.
Nombre:	getRolById($id)
Descripción:	Ejecuta la consulta que devuelve los datos de un rol, en el lenguaje que recibe como parámetro de entrada.
Nombre:	editRol($array)
Descripción:	Ejecuta la consulta que modifica los datos de un rol sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	existeRol($idrol)
Descripción:	Ejecuta la consulta que devuelve verdadero si existe en la base de datos un rol con el mismo identificador que recibe como parámetros de entrada y falso si no existe.

Nombre: Gestionar_Autenticación	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	verificarDatos($user, $pass, $code)
Descripción:	Verifica los datos del usuario y si están correctos le permite el acceso.

Nombre:	getDatosUser($user)
Descripción:	Devuelve los datos del usuario cuyo identificador recibe como parámetro de entrada.
Nombre:	changePass($array)
Descripción:	Permite cambiar la contraseña de un usuario.

Nombre: Autenticación_Factory	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getUser($user)
Descripción:	Ejecuta la consulta a la base de datos que devuelve el login de un usuario.
Nombre:	getPassByUser($user)
Descripción:	Ejecuta la consulta a la base de datos que devuelve la contraseña de un usuario.
Nombre:	getDatosUser($user)
Descripción:	Ejecuta la consulta a la base de datos que devuelve los datos del usuario cuyo identificador recibe como parámetro de entrada.
Nombre:	changePass($user, $pass)
Descripción:	Ejecuta la consulta a la base de datos que sustituye la contraseña de un usuario por la nueva que recibe como parámetro de entrada.

Nombre: Gestionar_Galería	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	addImagen($array)
Descripción:	Crea una nueva instancia en la tabla de la base de datos que guarda los datos de las imágenes con los valores que recibe como parámetro de entrada.
Nombre:	copyImgToServer($folderCat)
Descripción:	Copia las imágenes en una carpeta servidora que recibe como parámetro de entrada.
Nombre:	deleteImgFromServer($path)
Descripción:	Elimina la imagen de la carpeta servidora.
Nombre:	getLastImage()
Descripción:	Devuelve el identificador de la última imagen que ha sido guarda en la base de datos.
Nombre:	getImageById($idImg, $leng)
Descripción:	Devuelve los datos de una imagen en el lenguaje que recibe como parámetro de entrada.
Nombre:	editImage($array)
Descripción:	Modifica los datos de una imagen sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	deleteImage($idImg, $idCat)

Descripción:	Elimina los datos de una imagen cuyo identificador recibe como parámetro de entrada.
Nombre:	getImages($lang)
Descripción:	Devuelve los datos de las imágenes, en el lenguaje que recibe como parámetro de entrada.
Nombre:	getCategorias($lang)
Descripción:	Devuelve las categorías de las imágenes en el lenguaje que recibe como parámetro de entrada.

Nombre: Galería_Factory	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	addImage($idCat, $img)
Descripción:	Ejecuta la consulta que crea una nueva instancia en la tabla de la base de datos que guarda los datos de las imágenes con los valores que recibe como parámetro de entrada.
Nombre:	addImageLang($idImg, $idLeng, $desc)
Descripción:	Ejecuta la consulta que crea una nueva instancia en la tabla de la base de datos que guarda los datos de las imágenes en el lenguaje que recibe como parámetro de entrada.
Nombre:	getLasImage()
Descripción:	Ejecuta la consulta que devuelve el identificador de la última imagen que ha sido guarda en la base de datos.
Nombre:	getImageById($idImg, $leng)
Descripción:	Ejecuta la consulta que devuelve los datos de una imagen en el lenguaje que recibe como parámetro de entrada.
Nombre:	editImage($idleng, $idimg, $idcat, $img, $desc)
Descripción:	Ejecuta la consulta que modifica los datos de una imagen sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	deleteIamge($idImg)
Descripción:	Ejecuta la consulta que elimina los datos de una imagen cuyo identificador recibe como parámetro de entrada.
Nombre:	getImages($lang)
Descripción:	Ejecuta la consulta que devuelve los datos de las imágenes, en el lenguaje que recibe como parámetro de entrada.
Nombre:	getCategorias($lang)
Descripción:	Devuelve los datos de las imágenes, en el lenguaje que recibe como parámetro de entrada.
Nombre:	getCategorias($lang)
Descripción:	Ejecuta la consulta que devuelve las categorías de las imágenes en el lenguaje que recibe como parámetro de entrada.

Nombre: Gestioner_Inf_Sitio_Web	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	listarCategorias($lang)
Descripción:	Devuelve una lista con todas las categorías de los sitios web.
Nombre:	getIdCategoriaActiva()
Descripción:	Devuelve el identificador de la categoría activa.
Nombre:	getSitiosByCat($idCat, $lang)
Descripción:	Devuelve un listado de los sitios de la categoría que recibe como parámetro de entrada.

Nombre: Sitio_Web_Factory_Usuario	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	listarCategorias($lang)
Descripción:	Ejecuta la consulta a la base de datos que devuelve una lista con todas las categorías de los sitios web.
Nombre:	getIdCategoriaActiva()
Descripción:	Ejecuta la consulta a la base de datos que devuelve el identificador de la categoría activa.
Nombre:	getSitiosByCat($idCat, $lang)
Descripción:	Ejecuta la consulta a la base de datos que devuelve un listado de los sitios de la categoría que recibe como parámetro de entrada.

Nombre: Gestionar_Residencia	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getResidencia($lang)
Descripción:	Devuelve los datos de las residencias, en el lenguaje que recibe como parámetro de entrada.
Nombre:	addResidencia($array)
Descripción:	Crea una nueva instacia en la tabla de la base de datos que guarda los datos de las residencias con los valores que recibe como parámetro de entrada.
Nombre:	delResidencia($ip)
Descripción:	Elimina los datos de la residencia cuyo identificador recibe como parámetro de entrada.
Nombre:	getResidenciaByIp($ip,$lang)
Descripción:	Devuelve los datos de una residencia, en el lenguaje que recibe como

	parámetro de entrada.
Nombre:	editResidencia($array)
Descripción:	Modifica los datos de una residencia sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	existeResidencia($ip)
Descripción:	Devuelve verdadero si existe en la base de datos una residencia con el valor de IP que recibe como parámetros de entrada y falso si no existe.
Nombre:	existeResidenciaLeng($ipR, $idLeng)
Descripción:	Devuelve verdadero si existe en la base de datos una residencia con los valores que recibe como parámetros de entrada y falso si no existe.

Nombre: Residencia_Factory	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getResidencia($lang)
Descripción:	Ejecuta la consulta que devuelve los datos de las residencias, en el lenguaje que recibe como parámetro de entrada.
Nombre:	addResidencia($zona, $nivel, $ipR, $numero, $idImg)
Descripción:	Ejecuta la consulta que crea una nueva instancia en la tabla de la base de datos que guarda los datos de las residencias con los valores que recibe como parámetro de entrada.
Nombre:	addResidenciaLenguage($ipResidencia, $idLeng, $historia, $interiores, $direccion, $nombre)
Descripción:	Ejecuta la consulta que guarda los datos de las residencias en el lenguaje que recibe como parámetro de entrada.
Nombre:	delResidencia($ip)
Descripción:	Ejecuta la consulta que elimina los datos de la residencia cuyo identificador recibe como parámetro de entrada.
Nombre:	getResidenciaByIp($ip,$lang)
Descripción:	Ejecuta la consulta que devuelve los datos de una residencia, en el lenguaje que recibe como parámetro de entrada.
Nombre:	existeResidenciaLeng($ipR, $idLeng)
Descripción:	Ejecuta la consulta que devuelve verdadero si existe en la base de datos una residencia con los valores que recibe como parámetros de entrada y falso si no existe.
Nombre:	existeResidencia($ip)
Descripción:	Ejecuta la consulta que devuelve verdadero si existe en la base de datos una residencia con el valor de IP que recibe como parámetros de entrada y falso si no existe.
Nombre:	editResidencia($idLeng, $ipR, $zona, $nivel, $numero, $historia, $interiores, $direccion, $ipH, $nombre)
Descripción:	Ejecuta la consulta que modifica los datos de una residencia sustituyéndolos por los datos que recibe como parámetros de entrada.

Nombre: Gestionar_Insumo	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getInsumo($lang)
Descripción:	Devuelve los datos de los insumos, en el lenguaje que recibe como parámetro de entrada.
Nombre:	addInsumo($array)
Descripción:	Crea una nueva instacia en la tabla de la base de datos que guarda los datos de los insumos con los valores que recibe como parámetro de entrada.
Nombre:	delInsumo($id)
Descripción:	Elimina los datos del insumo cuyo identificador recibe como parámetro de entrada.
Nombre:	getInsumoById($id, $lang)
Descripción:	Devuelve los datos de un insumo, en el lenguaje que recibe como parámetro de entrada.
Nombre:	editInsumo($array)
Descripción:	Modifica los datos de un insumo sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	getCategorias()
Descripción:	Devuelve las categorías de los insumos.
Nombre:	existeInsumoLeng($idB, $idLeng)
Descripción:	Devuelve verdadero si existe en la base de datos un insumo con los valores que recibe como parámetros de entrada y falso si no existe.

Nombre: Insumo_Factory	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getInsumo($lang)
Descripción:	Ejecuta la consulta que devuelve los datos de los insumos, en el lenguaje que recibe como parámetro de entrada.
Nombre:	addInsumo($array)
Descripción:	Ejecuta la consulta que crea una nueva instancia en la tabla de la base de datos que guarda los datos de los insumos con los valores que recibe como parámetro de entrada.
Nombre:	delInsumo($id)
Descripción:	Ejecuta la consulta que elimina los datos del insumo cuyo identificador recibe como parámetro de entrada.
Nombre:	getInsumoById($id, $lang)
Descripción:	Ejecuta la consulta que devuelve los datos de un insumo, en el lenguaje que recibe como parámetro de entrada.
Nombre:	editInsumo($array)
Descripción:	Ejecuta la consulta que modifica los datos de un insumo sustituyéndolos por los datos que recibe como parámetros de entrada.

Nombre:	getCategorias()
Descripción:	Ejecuta la consulta que devuelve las categorías de los insumos.
Nombre:	existeInsumoLeng($idB, $idLeng)
Descripción:	Ejecuta la consulta que devuelve verdadero si existe en la base de datos un insumo con los valores que recibe como parámetros de entrada y falso si no existe.

Nombre: Menú_Factory	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	getIngredientes($lang)
Descripción:	Ejecuta la consulta que devuelve los ingredientes en el lenguaje que recibe como parámetro de entrada.
Nombre:	getIngredienteMenu($lang)
Descripción:	Ejecuta la consulta que devuelve el menú con sus ingredientes en el lenguaje que recibe como parámetro de entrada.
Nombre:	listIngredientes($lang)
Descripción:	Ejecuta la consulta que devuelve una lista de ingredientes en el lenguaje que recibe como parámetro de entrada.
Nombre:	getDiaSemana($lang)
Descripción:	Ejecuta la consulta que devuelve el día de la semana en el lenguaje que recibe como parámetro de entrada.
Nombre:	getDiaSemanaMenu($lang)
Descripción:	Ejecuta la consulta que devuelve el día de la semana de cada de menú en el lenguaje que recibe como parámetro de entrada.
Nombre:	addMenu($array)
Descripción:	Ejecuta la consulta que crea una nueva instancia en la tabla de la base de datos que guarda los datos del menú con los valores que recibe como parámetro de entrada.
Nombre:	getIngredienteById($itemid, $lang)
Descripción:	Ejecuta la consulta que devuelve los datos de un ingrediente, en el lenguaje que recibe como parámetro de entrada.
Nombre:	editMenu($array)
Descripción:	Ejecuta la consulta que modifica los datos del menú sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	addIngrediente($array)
Descripción:	Ejecuta la consulta que crea una nueva instancia en la tabla de la base de datos que guarda los datos de los ingredientes con los valores que recibe como parámetro de entrada.
Nombre:	delIngrediente($id)
Descripción:	Ejecuta la consulta que elimina los datos del ingrediente cuyo identificador recibe como parámetro de entrada.
Nombre:	editIngrediente($array)
Descripción:	Ejecuta la consulta que modifica los datos de un ingrediente sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	getCategorias($tipo)
Descripción:	Ejecuta la consulta que devuelve las categorías del menú en el tipo que recibe como parámetro de entrada.
Nombre:	
Descripción:	

Nombre: Gestionar_Persona	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getPersona($lang)
Descripción:	Devuelve los datos de las personas, en el lenguaje que recibe como parámetro de entrada.
Nombre:	addPersona($array)
Descripción:	Crea una nueva instancia en la tabla de la base de datos que guarda los datos de las personas con los valores que recibe como parámetro de entrada.
Nombre:	delPersona($idP)
Descripción:	Elimina los datos de un apersona cuyo identificador recibe como parámetro de entrada.
Nombre:	getPersonaById($idP, $lang)
Descripción:	Devuelve los datos de una persona, en el lenguaje que recibe como parámetro de entrada.
Nombre:	editPersona($array)
Descripción:	Modifica los datos de una persona sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	getCategorias()
Descripción:	Devuelve las categorías de las personas.
Nombre:	getResidencias()
Descripción:	Devuelve la residencia que está destinada a las personas.
Nombre:	copyFoto()
Descripción:	Devuelve la foto correspondiente a cada persona.
Nombre:	deleteFoto($img)
Descripción:	Elimina la foto de una persona cuyo identificador recibe como parámetro de entrada.

Nombre: Persona_Factory	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getPersona($lang)
Descripción:	Ejecuta la consulta que devuelve los datos de las personas, en el lenguaje que recibe como parámetro de entrada.
Nombre:	addPersona($array)
Descripción:	Ejecuta la consulta que crea una nueva instancia en la tabla de la base de datos que guarda los datos de las personas con los valores que recibe como parámetro de entrada.
Nombre:	delPersona($idP)
Descripción:	Ejecuta la consulta que elimina los datos de un apersona cuyo identificador recibe como parámetro de entrada.
Nombre:	getPersonaById($idP, $lang)

Descripción:	Ejecuta la consulta que devuelve los datos de una persona, en el lenguaje
	que recibe como parámetro de entrada.
Nombre:	editPersona($array)
Descripción:	Ejecuta la consulta que modifica los datos de una persona sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	getCategorias()
Descripción:	Ejecuta la consulta que devuelve las categorías de las personas.
Nombre:	getResidencias()
Descripción:	Ejecuta la consulta que devuelve la residencia que está destinada a las personas.
Nombre:	copyFoto()
Descripción:	Ejecuta la consulta que devuelve la foto correspondiente a cada persona.
Nombre:	deleteFoto($img)
Descripción:	Ejecuta la consulta que elimina la foto de una persona cuyo identificador recibe como parámetro de entrada.

Nombre: Gestionar_Usuario	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getUsuarios()
Descripción:	Devuelve los datos de los usuarios.
Nombre:	addUsuario($array)
Descripción:	Crea una nueva instancia en la tabla de la base de datos que guarda los datos de los usuarios con los valores que recibe como parámetro de entrada.
Nombre:	delUsuario($id))
Descripción:	Elimina los datos de un usuario cuyo identificador recibe como parámetro de entrada.
Nombre:	getUsurioById($id)
Descripción:	Devuelve los datos de un usuario, en el lenguaje que recibe como parámetro de entrada.
Nombre:	editUsuario($array)
Descripción:	Modifica los datos de un usuario sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	existeUsuario($user)
Descripción:	Devuelve verdadero si existe en la base de datos un usuario con el dato que recibe como parámetros de entrada y falso si no existe.
Nombre:	editPermisos($array)
Descripción:	Modifica los datos de los permisos de los usuarios sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	getModulos($idU)
Descripción:	Devuelve el módulo correspondiente a los usuarios por el dato que recibe como parámetro de entrada.
Nombre:	getDptos()
Descripción:	Devuelve el departamento de los usuarios.

Nombre: Usuario_Factory	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getUsuarios()
Descripción:	Ejecuta la consulta que devuelve los datos de los usuarios.
Nombre:	addUsuario($array)
Descripción:	Ejecuta la consulta que crea una nueva instancia en la tabla de la base de datos que guarda los datos de los usuarios con los valores que recibe como parámetro de entrada.
Nombre:	delUsuario($id))
Descripción:	Ejecuta la consulta que elimina los datos de un usuario cuyo identificador recibe como parámetro de entrada.
Nombre:	getUsurioById($id)
Descripción:	Ejecuta la consulta que devuelve los datos de un usuario, en el lenguaje que recibe como parámetro de entrada.
Nombre:	editUsuario($array)
Descripción:	Ejecuta la consulta que modifica los datos de un usuario sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	existeUsuario($user)
Descripción:	Ejecuta la consulta que devuelve verdadero si existe en la base de datos un usuario con el dato que recibe como parámetros de entrada y falso si no existe.
Nombre:	editPermisos($array)
Descripción:	Ejecuta la consulta que modifica los datos de los permisos de los usuarios sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	getModulos($idU)
Descripción:	Ejecuta la consulta que devuelve el módulo correspondiente a los usuarios por el dato que recibe como parámetro de entrada.
Nombre:	getModulosAcceso($idU)
Descripción:	Ejecuta la consulta que da el acceso a los módulos en dependencia del usuario que recibe como parámetro de entrada.
Nombre:	getDptos()
Descripción:	Ejecuta la consulta que devuelve el departamento de los usuarios.

Nombre: Gestionar_Solicitud_Lavandería.	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getPedidos()
Descripción:	Devuelve las solicitudes de lavandería.
Nombre:	deletePedido($id)
Descripción:	Elimina los datos de la solicitud cuyo identificador recibe como parámetro de entrada.

Nombre: Lavandería_Factory	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getPedidos()
Descripción:	Ejecuta la consulta que devuelve las solicitudes de la lavandería.
Nombre:	deletePedido($id)

Nombre: Gestionar_Opinión	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getOpiniones($dpto)
Descripción: Nombre: Descripción	Devuelve los datos de las opiniones de un servicio. deleteOpinión($idop) Elimina la opinión cuyo identificador recibe como parámetro de entrada.

Nombre: Opinión_Factory	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getOpiniones($dpto)
Descripción: Nombre: Descripción	Ejecuta la consulta que devuelve los datos de las opiniones de un servicio. deleteOpinión($idop) Ejecuta la consulta que elimina la opinión cuyo identificador recibe como parámetro de entrada.

Nombre: Gestionar_Obra_Arte	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getObras()
Descripción:	Devuelve los datos de las obras de arte.
Nombre:	addObra($array)
Descripción:	Crea una nueva instancia en la tabla de la base de datos que guarda los datos de las obras de arte con los valores que recibe como parámetro de entrada.

Nombre:	delObra($id)
Descripción:	Elimina los datos de una obra arte cuyo identificador recibe como parámetro de entrada.
Nombre:	getObraById($id)
Descripción:	Devuelve los datos de una obra arte, en el lenguaje que recibe como parámetro de entrada.
Nombre:	editObra($array)
Descripción:	Modifica los datos de una obra arte sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	existeObraLeng($idB, $idLeng)
Descripción:	Devuelve verdadero si existe en la base de datos una obra de arte con los valores que recibe como parámetros de entrada y falso si no existe.
Nombre:	getDenominacion()
Descripción:	Devuelve los datos de la denominación de las obras de arte.
Nombre:	getManifestacion()
Descripción:	Devuelve los datos de la manifestación de las obras de arte.

Nombre: Obra_Arte_Factory	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getObras()
Descripción:	Ejecuta la consulta que devuelve los datos de las obras de arte.
Nombre:	addObra($array)
Descripción:	Ejecuta la consulta que crea una nueva instancia en la tabla de la base de datos que guarda los datos de las obras de arte con los valores que recibe como parámetro de entrada.
Nombre:	delObra($id)
Descripción:	Ejecuta la consulta que elimina los datos de una obra arte cuyo identificador recibe como parámetro de entrada.
Nombre:	getObraById($id)
Descripción:	Ejecuta la consulta que devuelve los datos de una obra arte, en el lenguaje que recibe como parámetro de entrada.
Nombre:	editObra($array)
Descripción:	Ejecuta la consulta que modifica los datos de una obra arte sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	existeObraLeng($idB, $idLeng)
Descripción:	Ejecuta la consulta que devuelve verdadero si existe en la base de datos una obra de arte con los valores que recibe como parámetros de entrada y falso si no existe.
Nombre:	getDenominacion()
Descripción:	Ejecuta la consulta que devuelve los datos de la denominación de las obras de arte.
Nombre:	getManifestacion()
Descripción:	Ejecuta la consulta que devuelve los datos de la manifestación de las obras de arte.

Nombre: Gestionar_Rol	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getRol()
Descripción:	Devuelve los datos de los roles.
Nombre:	addRol($array)
Descripción:	Crea una nueva instancia en la tabla de la base de datos que guarda los datos de los roles con los valores que recibe como parámetro de entrada.
Nombre:	delRol($id)
Descripción:	Elimina los datos de un rol cuyo identificador recibe como parámetro de entrada.
Nombre:	getRolById($id)
Descripción:	Devuelve los datos de un rol, en el lenguaje que recibe como parámetro de entrada.
Nombre:	editRol($array)
Descripción:	Modifica los datos de un rol sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	existeRolLeng($idB, $idLeng)
Descripción:	Devuelve verdadero si existe en la base de datos una rol con los valores que recibe como parámetros de entrada y falso si no existe.

Nombre: Rol_Factory	
Clase Controladora.	
Atributo	**Tipo**
Métodos:	
Nombre:	__construct()
Descripción:	Construye la clase.
Nombre:	getRol()
Descripción:	Ejecuta la consulta que devuelve los datos de los roles.
Nombre:	addRol($array)
Descripción:	Ejecuta la consulta que crea una nueva instancia en la tabla de la base de datos que guarda los datos de los roles con los valores que recibe como parámetro de entrada.
Nombre:	delRol($id)
Descripción:	Ejecuta la consulta que elimina los datos de un rol cuyo identificador recibe como parámetro de entrada.
Nombre:	getRolById($id)
Descripción:	Ejecuta la consulta que devuelve los datos de un rol, en el lenguaje que recibe como parámetro de entrada.
Nombre:	editRol($array)
Descripción:	Ejecuta la consulta que modifica los datos de un rol sustituyéndolos por los datos que recibe como parámetros de entrada.
Nombre:	existeRolLeng($idB, $idLeng)
Descripción:	Ejecuta la consulta que devuelve verdadero si existe en la base de datos una rol con los valores que recibe como parámetros de entrada y falso si no existe.

Nombre: Rol	
Clase Entidad	
Atributo	**Tipo**
idrol	integer
rol	varchar
modulo	integer
identificador	integer
path	varchar
activa	integer
Métodos:	

Nombre: Encuesta	
Clase Entidad	
Atributo	**Tipo**
idencuesta	integer
fecha	date
fecha_pub	date
fecha_exp	date
respuesta	varchar
activa	integer
Métodos:	

Nombre: Opinión	
Clase Entidad	
Atributo	**Tipo**
idopinión	integer
opinió	varchar
fecha	date
nombre_envía	varchar
Métodos:	

Nombre: Residencia	
Clase Entidad	
Atributo	**Tipo**
ipresidencia	varchar
residencia	varchar
zona	varchar
número	varchar
nivel	varchar
historia	varchar
interiores	varchar
dirección	varchar
Métodos:	

Nombre: Obra	
Clase Entidad	
Atributo	**Tipo**
idobra	integer
denominación	varchar
manifestación	varchar
técnica	varchar
medida	real
soporte	varchar
descripción	varchar
estado	varchar
valor	real
color	varchar
grado	varchar
origen	varchar
tasación	real
ubicación	varchar
inventario	varchar
temática	varchar
título	varchar
catalogador	varchar
fecha	date
autores	varchar
lugar	varchar
escuela	varchar
editora	varchar
forma	varchar
imagen_file	intiger
imagen_post	intiger
época	varchar
Métodos:	

Nombre: Servicio	
Clase Entidad	
Atributo	**Tipo**
idservicio	integer
servicio	varchar
Métodos:	

Nombre: Imagen	
Clase Entidad	
Atributo	**Tipo**
idimagen	integer
imagen	integer
categoría	varchar
descripción	varchar
Métodos:	

Nombre: Menú	
Clase Entidad	
Atributo	**Tipo**
idmenú	integer
fecha	date
categoría	varchar
ingrediente	varchar
tipo	integer
orden	varchar
día_semana	varchar
Métodos:	

Nombre: Persona	
Clase Entidad	
Atributo	**Tipo**
idpersona	integer
nombre	varchar
descripción	varchar
activa	integer
Métodos:	

Nombre: Insumo	
Clase Entidad	
Atributo	**Tipo**
idinsumo	integer
insumo	varchar
categoría	varchar
descripción	varchar
activa	integer
Métodos:	

Nombre: Usuario	
Clase Entidad	
Atributo	**Tipo**
idusuario	integer
nombre	varchar
apellidos	varchar
departamento	varchar
correo	varchar
login	varchar
pass	varchar
rol	integer
activa	integer
Métodos:	

Nombre: Sitio	
Clase Entidad	
Atributo	**Tipo**
idsitio	integer
título	varchar
categoría	varchar
descripción	varchar
imagen	integer
link	varchar
activa	integer
Métodos:	

Nombre: Solicitud	
Clase Entidad	
Atributo	**Tipo**
idsolicitud	integer
nombre_solicita	varchar
hora_solicita	varchar
hora_entrega	varchar
fecha_solicita	date
fecha_entrega	date
habitación	varchar
tipo_servicio	varchar
Métodos:	

3.6 Diagrama Entidad Relación.

El Diagrama de Entidad Relación posee una relativa simplicidad, representa la información mediante una colección de tablas bidimensionales.

El diagrama se encuentra fragmentado debido a su extenso tamaño.

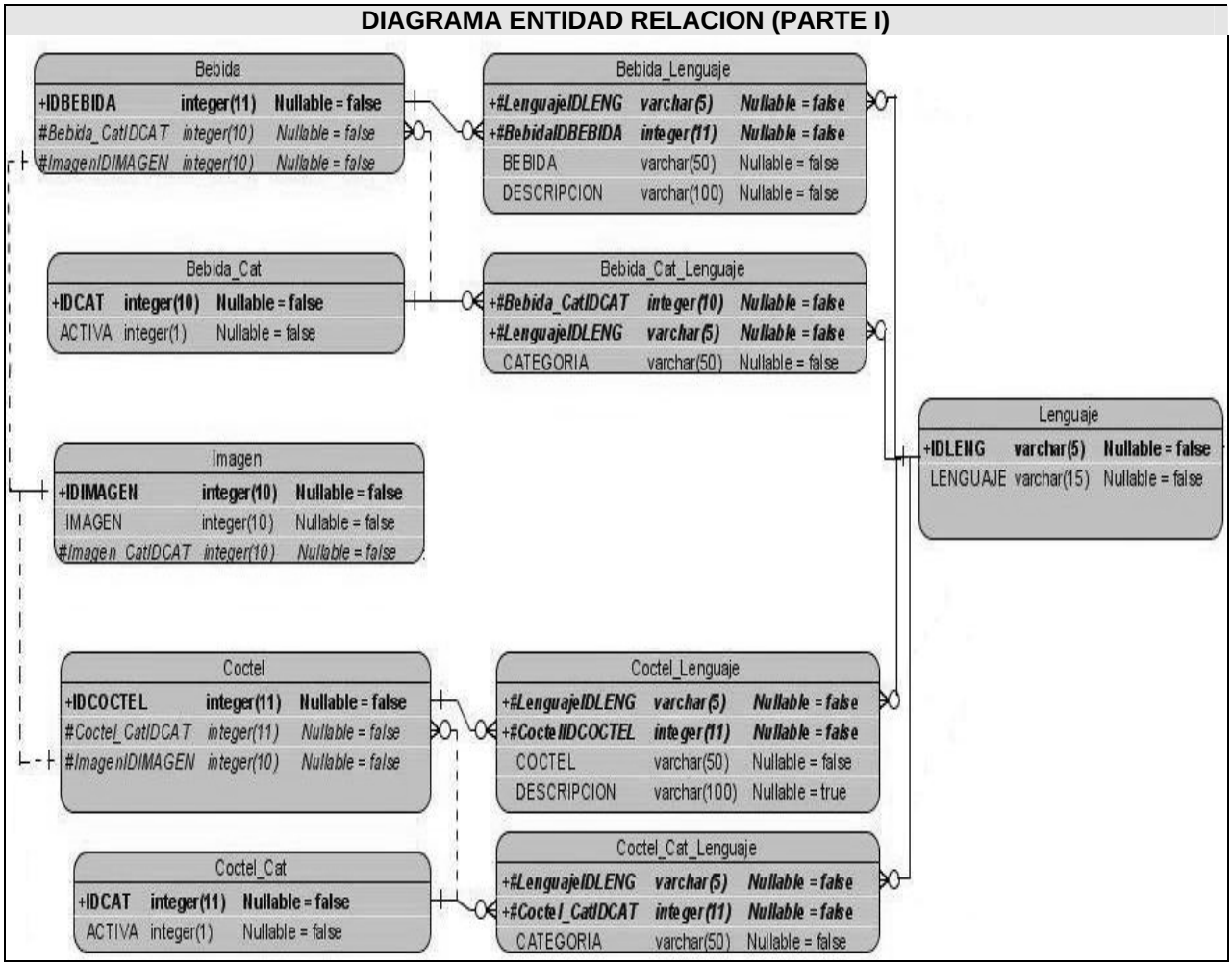


DIAGRAMA ENTIDAD RELACION (PARTE II)

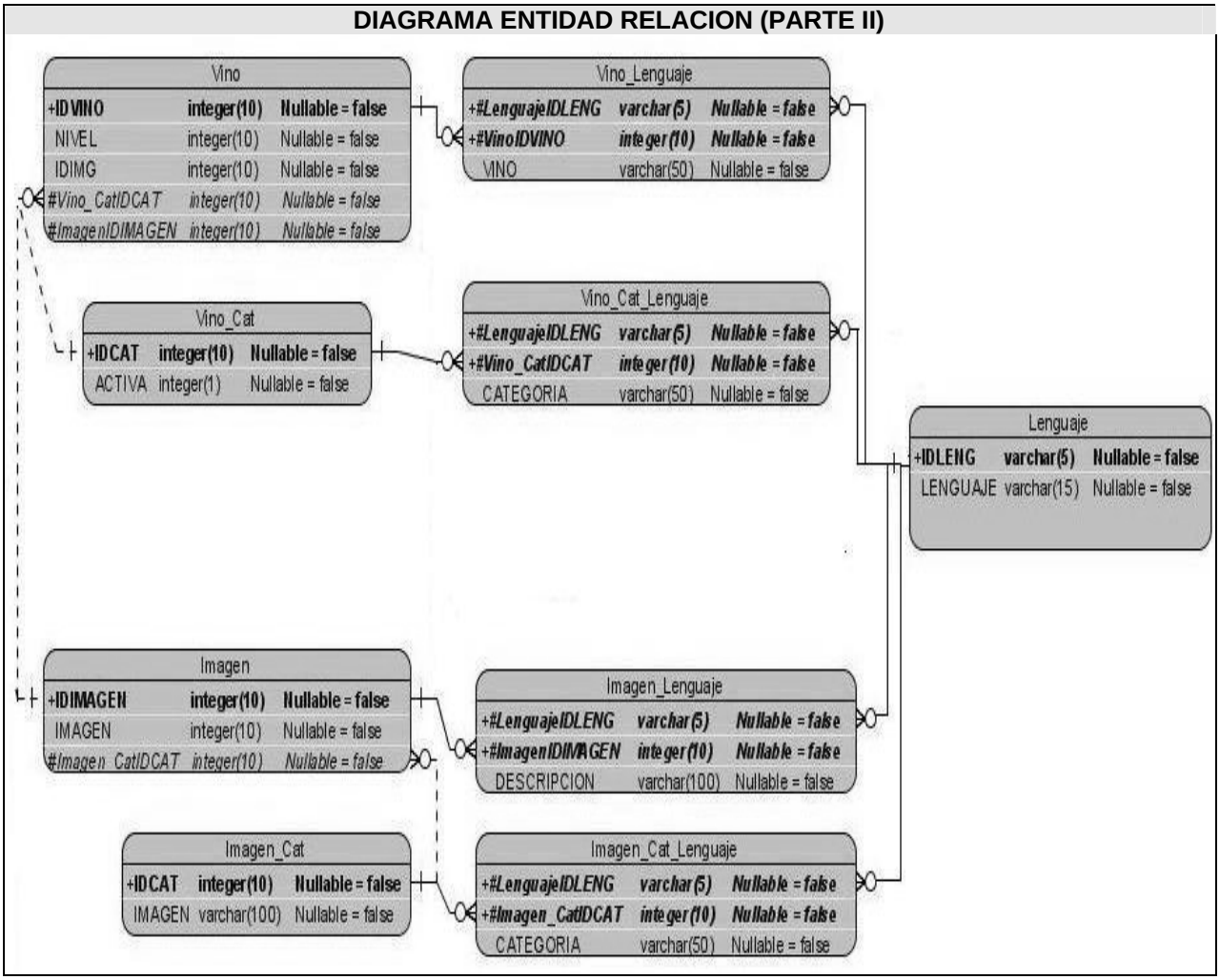


DIAGRAMA ENTIDAD RELACION (PARTE III)

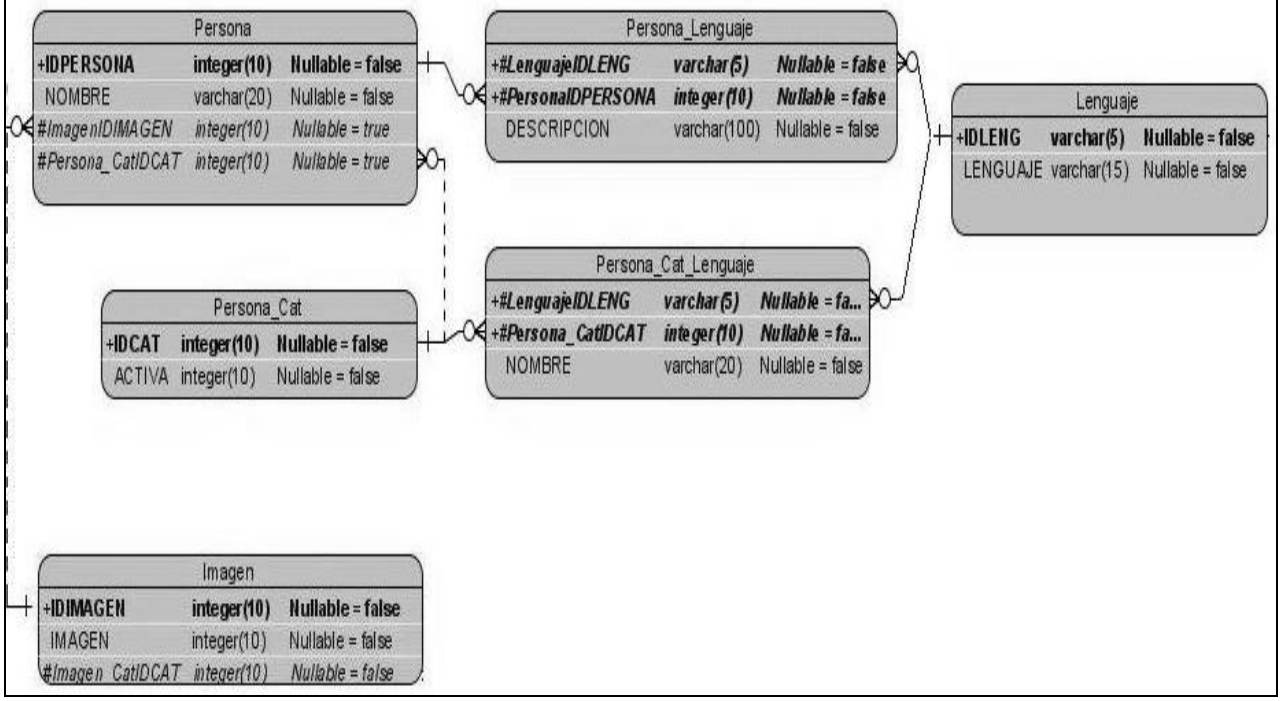


DIAGRAMA ENTIDAD RELACION (PARTE IV)

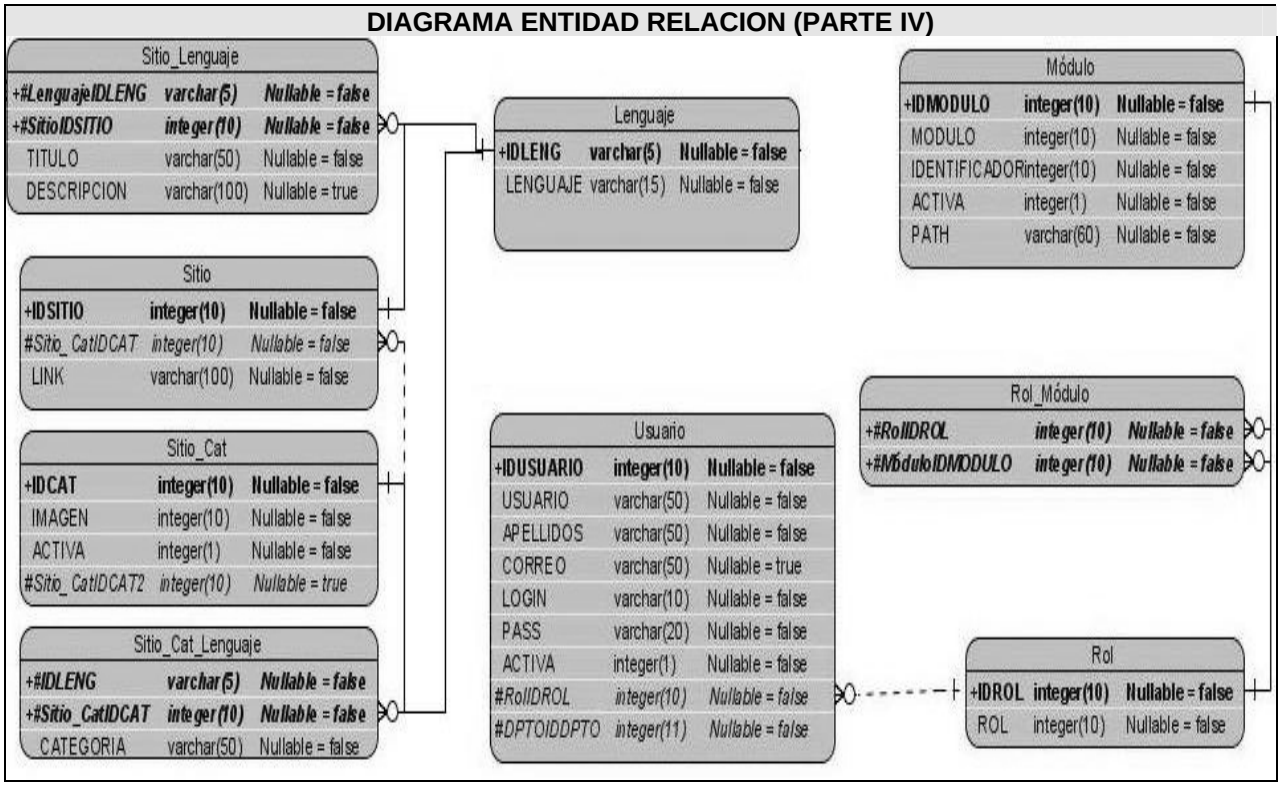


DIAGRAMA ENTIDAD RELACION (PARTE V)

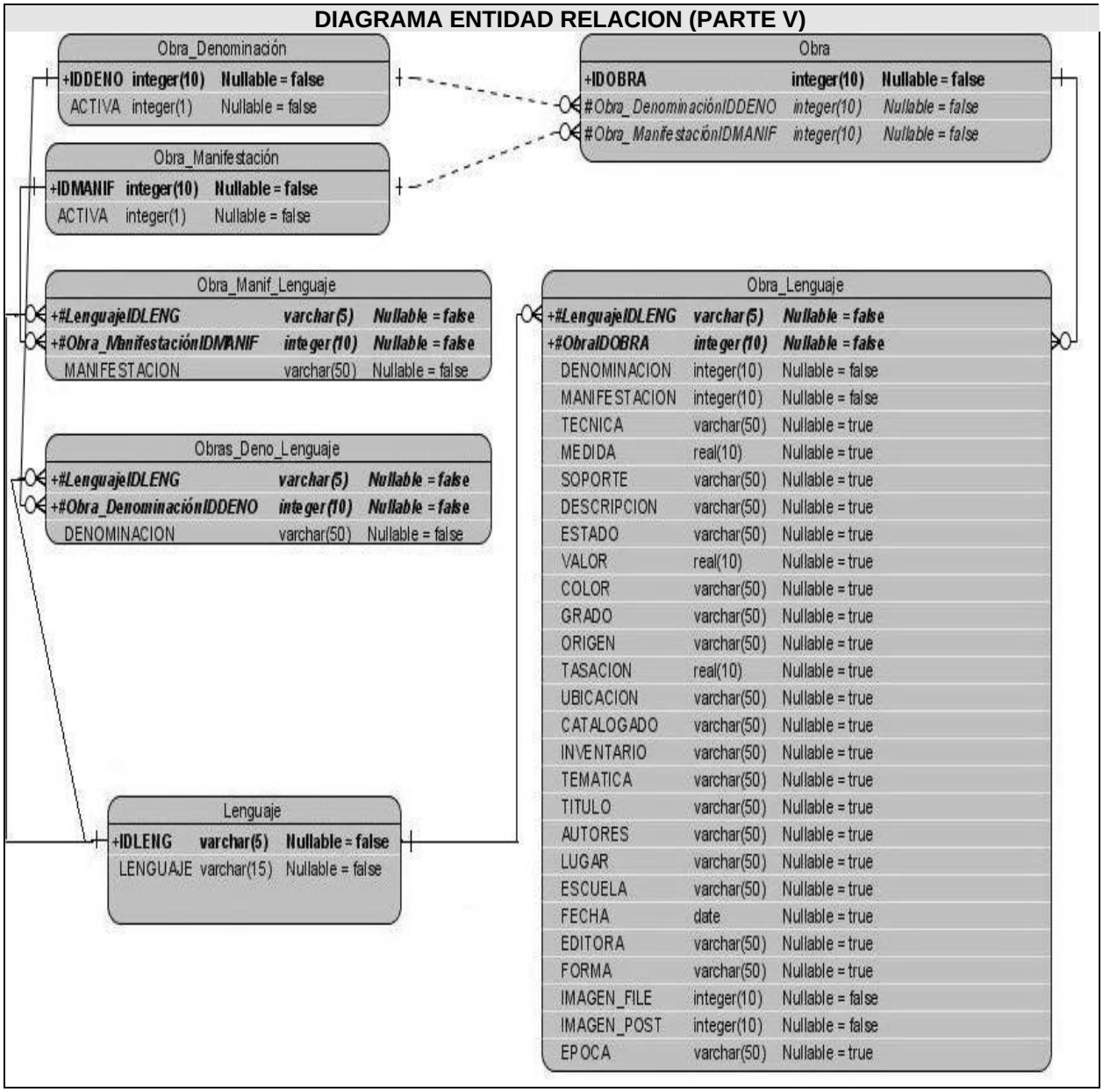


DIAGRAMA ENTIDAD RELACION (PARTE VI)

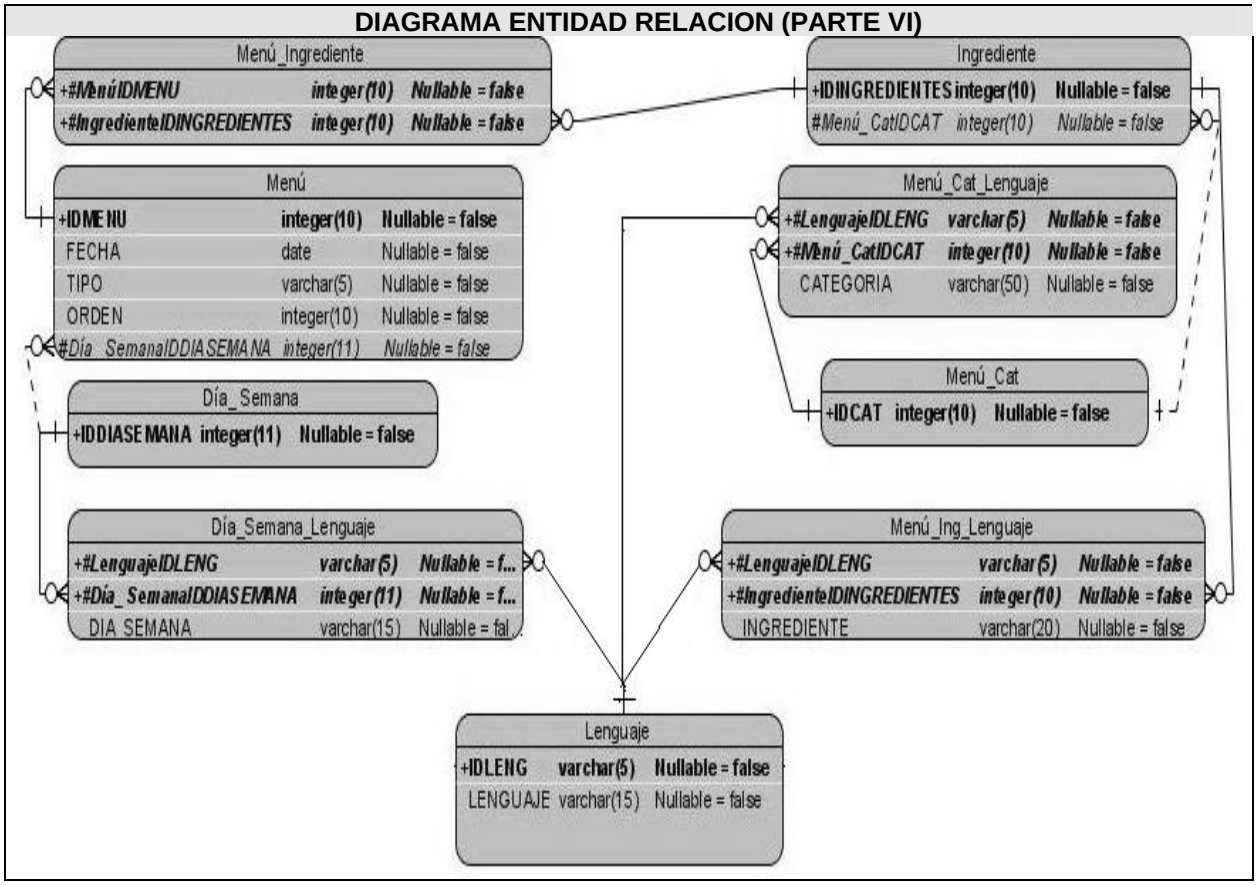


DIAGRAMA ENTIDAD RELACION (PARTE VII)

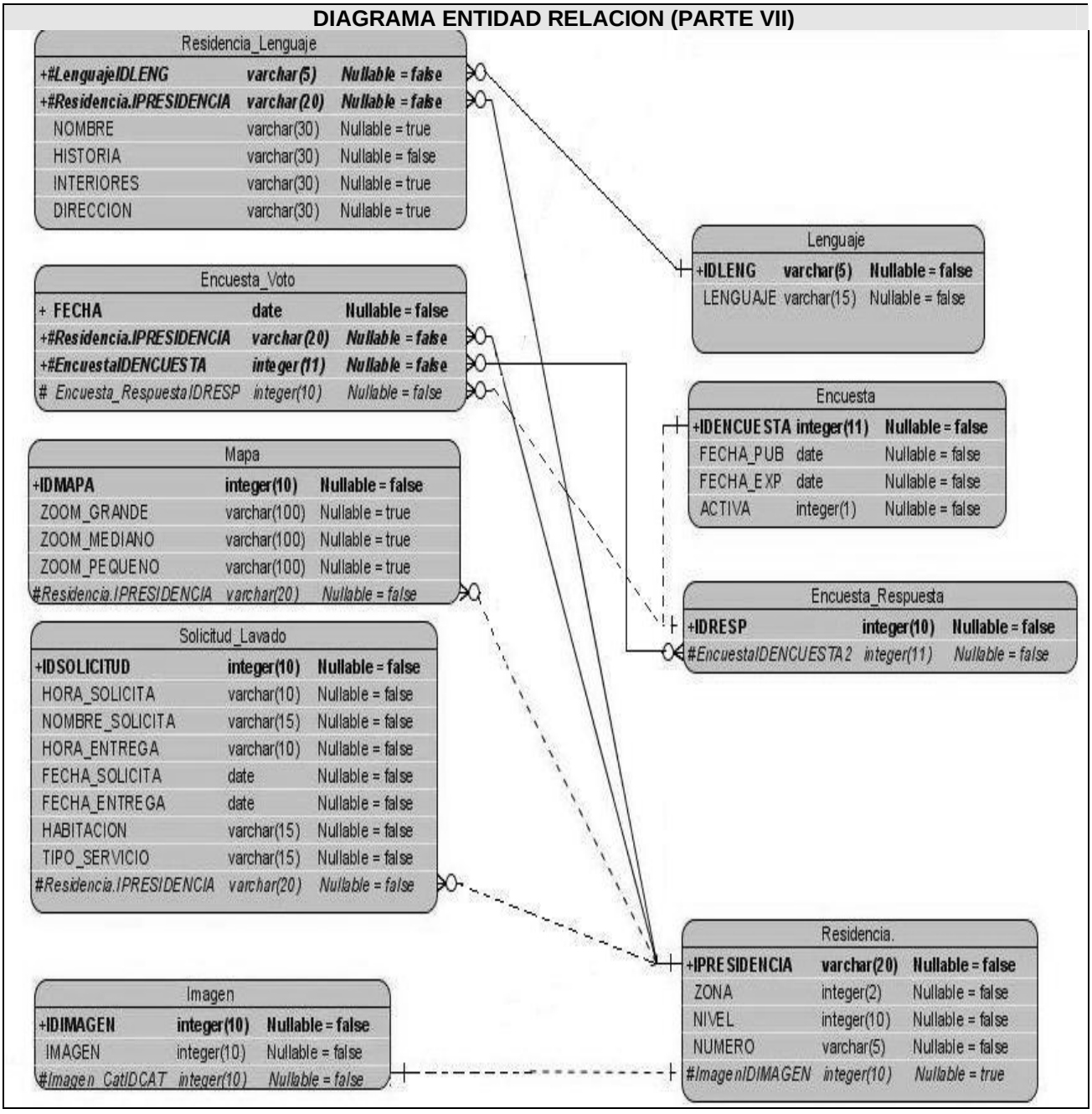
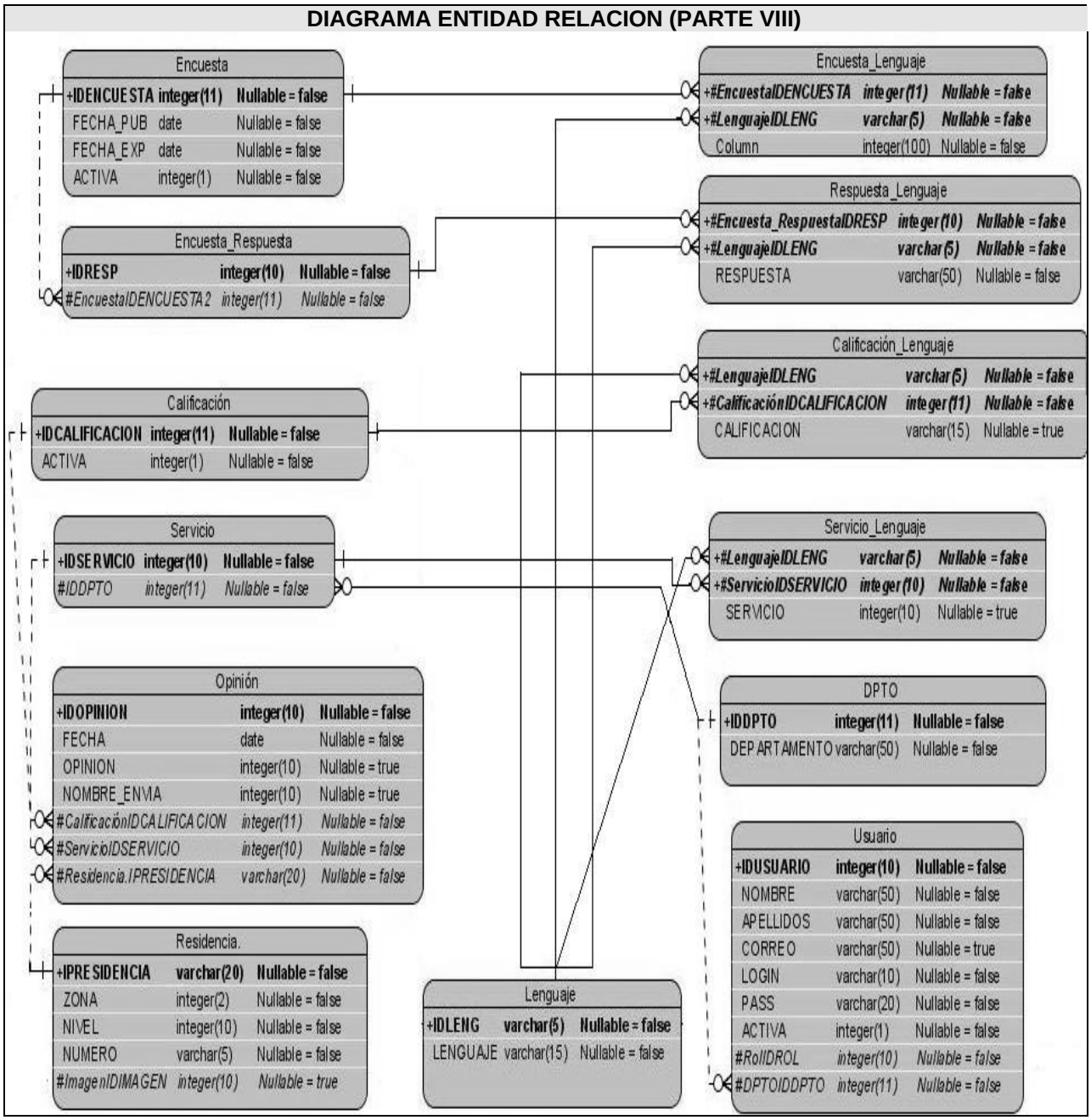


DIAGRAMA ENTIDAD RELACION (PARTE VIII)



Para observar el Diagramas Entidad Relación con la unión de todas sus partes remítase a los anexos en la versión digital contenida en el CD adjunto.

Conclusiones

- Se confeccionaron los diagramas de clases de análisis y de interacción correspondientes a cada caso de uso del sistema.
- Se definió el diseño de la Intranet a través de los diagrama de clases del diseño para cada caso de uso del sistema.
- Se especificaron las características de los patrones de diseño aplicados.
- Se describieron las clases del diseño especificando sus atributos y métodos.
- Se conformó el diagrama de entidad-relación entre las clases persistentes.

Conclusiones

- El estudio del entorno de trabajo de los usuarios y de los procesos internos de la entidad es vital para el desarrollo de un sistema informático.
- Se hacen imprescindibles para facilitar los procesos de gestión de la información en todo tipo de organizaciones el desarrollo de las aplicaciones Web como parte de las nuevas tecnologías de la informática y las comunicaciones. Lo cual tributa a la eficiencia en la labor de la institución.
- Resulta conveniente el uso de una metodología que guíe los pasos del desarrollo de la aplicación.
- El análisis previo al diseño permite conseguir una comprensión más precisa de los requisitos, propiciando una visión general del sistema.
- Utilizar los patrones de diseño resultó ventajoso pues permitió adaptar el conocimiento acumulado a las características propias del sistema y la creación de un diseño basado en clases reutilizables.
- El diseño de la Intranet permitió traducir los requisitos a una especificación que describe cómo implementar el sistema.
- El diseño de la Intranet crea una entrada apropiada y un punto de partida para actividades de implementación, pues permite descomponer los trabajos de implementación en partes más manejables.

Recomendaciones

- Se sugiere confeccionar una arquitectura de información que organice y clasifique los contenidos de manera que resulten comprensibles y atractivos a los usuarios.
- Se recomienda incluir al diseño un buscador de contenido de la Intranet para facilitar el manejo de la información.
- Se recomienda añadir el diseño de una sección que presente los programas de los eventos a los cuales deben asistir los huéspedes y que pueda ser actualizado desde el módulo de administración.

Bibliografía

- C. Costilla. "Características Objetos Relacionales del SGBD Oracle". 2000.
- Craig Larman “UML y Patrones. Introducción al Análisis y Diseño Orientado a objetos” Parte I y II. 2004. [Consulta: 3 de junio 2007].
- Electronic sources [en línea]: "Solmeliá". http://www.solmelia.com/solNew/home/jsp/C_Home.jsp?cid=BUS002:REG:marcas. [Consulta: 3 dic.2006].
- Electronic sources [en línea]: ”Hotel Nacional de Cuba”. http://www.hotelnacionaldecuba.com/sp/home.asp. [Consulta: 3 dic.2006].
- Electronic sources [en línea]: “Hotel La Costa”. http://www.lacostahotel.com/. [Consulta: 3 dic.2006].
- Electronic sources [en línea]: “Alcora Hoteles”. http://www.alcorahoteles.com/. [Consulta: 3 dic.2006].
- Electronic sources [en línea]: "¿PHP, Python, ASP, Perl o JSP?” http://www.mononeurona.org/index.php?idp=150. [Consulta: 15 ene 2007].
- Electronic sources [en línea]: "Desarrollo Web con PHP y MySQL". http://bibliodoc.uci.cu/pdf/reg02138.pdf [Consulta: 18 ene 2007].
- Electronic Book : "Programación de aplicaciones Web con JavaScript". Jerry Bradenbaugh [Consulta: 25 ene 2007].
- Electronic sources [en línea]: " Aprenda Programción en Microsoft SQL Server 2000 Ya". http://bibliodoc.uci.cu/pdf/reg01713.pdf. [Consulta: 28 ene 2007].
- Electronic sources [en línea]: "Visión general de las nuevas funcionalidades de Apache". http://httpd.apache.org/docs/2.0/es/new_features_2_0.html. [Consulta: 2 feb 2007].
- Electronic sources [en línea]: Camacho.M, Febles J, Orue M. **"**Experiencias de la Aplicación de la Ingeniería de Software en Sistema de Gestión. Revista cubana de informática medica**"**. [Consulta: 4 feb 2007]**.**
- Electronic sources [en línea]: "Visual Paradigm for UML" http://www.fileheaven.com/descargar/visual-paradigm-for-uml-community-edition-no-install/32084.htm. [Consulta: 8 feb 2007]**.**
- Electronic sources [en línea]: Fowler M, Scientist C. "La Nueva Metodología". http://www.programacionextrema.org/articulos/newMethodology.es.html#tth_sEc5.1. [Consulta: 8 feb 2007].
- Fuentes.E. Electronic sources [en línea]: "Internet, Intranets, Extranets." in http://www.ucm.es/info/multidoc/multidoc/revista/cuad6-7/eulalia.htm. [Consulta: 12 dic 2006].
- Gilfillan, L: “La Biblia de MySQL“. 2007.
- Ivar Jacobson, Grandy Booch, James Rumbaugh. '"“El proceso Unificado de Desarrollo de Software“. Volumen I y II. 2004. [Consulta: 10 abril 2007].
- Larman Craig.”UML y Patrones. Intoducción al Análisis y Diseño Orientado a Objetos”. Parte I. 2004.
- Roger S Pressman. “Ingienería de Software. Un Enfoque Práctico”. Parte I y II. 2005.
- Wendy Boggs, Michael Boggs. “UML with Rational Rose”. 2002.

Referencias Bibliográficas

1. SÁNCHEZ, J.M.L., "Intranet Corporativa bajo un entorno LAMP". in http://www.lcc.uma.es/pfc/351.pdf.

2. Powel, T.A., "Diseño Web". in http://bibliodoc.uci.cu/pdf/reg01293.pdf.

3. J. Rumbaugh, G.B.e.I.J., "El lenguaje Unificado de Modelado . Manual de Referencia".

4. Bradenbaugh, J., "Programación de aplicaciones Web con JavaScript". in http://bibliodoc.uci.cu/pdf/reg00009.pdf.

5. Vázquez, J.A.G., "Desarrollo Web con PHP y MySQL". in http://bibliodoc.uci.cu/pdf/reg02138.pdf.

6. Costilla, C., "Características Objetos Relacionales del SGBD Oracle". 2000. p. 2.

7. "Sobre PostgreSQL", in http://www.postgresql.org/about/. 2007.

8. "What is The Apache HTTP Server Proyect", in http://httpd.apache.org/ABOUT_APACHE.html.

9. Ivar Jacobson, G.B., James Rumbaugh, “El proceso Unificado de Desarrollo de Software“.1999.

10. "Herramienta CASE". 2007 [cited; Available from: http://es.wikipedia.org/wiki/Herramienta_CASE.

11. Caballero, I., "Visual Paradigm", in http://alarcos.inf-cr.uclm.es/per/fgarcia/isoftware/doc/LabTr1_VP.pdf. 2007.

Glosario de Términos

World Wide Web: También llamada WWW o Telaraña mundial. Es un sistema de comunicación de hipertexto que funciona sobre Internet.
Se referencia en la Introducción.

HTML : Acrónimo en inglés de Hyper Text Markup Language (Lenguaje de Marcación de Hipertexto), es un lenguaje de etiquetas diseñado para estructurar textos y presentarlos en forma de hipertexto, que es el formato estándar de las páginas Web.
Se referencia en la Fundamentación Teórica.

XHTML: Acrónimo inglés de Extensible Hyper Text Markup Language (Lenguaje Extensible de Marcado de Hipertexto), desarrollado por el Consorcio Mundial de Web (W3C). Es la versión XML de HTML, por lo que tiene, básicamente, las mismas funcionalidades, pero cumple las especificaciones, más estrictas, de XML.
Se referencia en el Capítulo 1, epígrafe 1.1.

Wiki: Es un sitio Web colaborativo llevado adelante por el trabajo colectivo de muchos autores. Le permite a cualquier usuario editar sus contenidos, aunque hayan sido creados por otros autores.
Se referencia en el Capítulo 1, epígrafe 1.1.

Weblog: Es un sitio Web donde se recopilan cronológicamente mensajes de uno o varios autores, sobre una temática en particular siempre conservando el autor la libertad de dejar publicado lo que crea pertinente.
Se referencia en el Capítulo 1, epígrafe 1.1.

TCP/IP: Acrónimo inglés de Transmission Control Protocol/Internet Protocol (Protocolo de Control de Transmisión/Protocolo de Internet). Conjunto básico de protocolos de comunicación de redes, que permiten la transmisión de información en redes de computadoras.
Se referencia en el Capítulo 1, epígrafe 1.2.

Firewall: En español, Cortafuegos, es un sistema de seguridad encargado de proteger una red de área local de accesos no autorizados desde una WAN a la que esté conectada, básicamente consiste en un filtro que mira la identidad de los paquetes y rechaza todos aquellos que no estén autorizados o correctamente identificados.
Se referencia en el Capítulo 1, epígrafe 1.2.

HTTP: Acrónimo inglés de HyperText Transmission Protocol (Protocolo de Transferencia de Hipertexto). Protocolo para transferir archivos o documentos hipertexto a través de la red. Se basa en una arquitectura cliente/servidor.
Se referencia en el Capítulo 1, epígrafe 1.3.

OMG: Acrónimo inglés de Object Management Group (Grupo de Administración de Objetos). Es un consorcio dedicado al cuidado y el establecimiento de diversos estándares de tecnologías orientadas a objetos, tales como UML, XMI, CORBA.
Se referencia en el Capítulo 1, epígrafe 1.4.

Cookies: Pequeños archivos de texto que un servidor Web almacena en el ordenador del usuario, para guardar información sobre éste, como un número de identificación, una contraseña, sus preferencias o cuántas veces ha visitado el sitio el usuario. Un sitio Web puede tener acceso a la información del cookie siempre que el usuario se conecta al servidor.
Se referencia en el Capítulo 1, epígrafe 1.4.2.

RDBMS: Acrónimo en ingles de Relational Data Base Manager System. Es un Sistema Administrador de Bases de Datos Relacionales.
Se referencia en el Capítulo 1, epígrafe 1.5.

ODBC: Acrónimo en ingles de Open Data Base Connectivity. Es un estándar de acceso a bases de datos desarrollado por Microsoft Corporation, el objetivo de ODBC es hacer posible el acceso a cualquier dato de cualquier aplicación, sin importar qué sistema gestor de bases de datos almacene los datos.
Se referencia en el Capítulo 1, epígrafe 1.5.

TCL: Acrónimo en ingles de Tool Command Language. Es un lenguaje de script creado que se usa principalmente en programas rápidos, aplicaciones "script", entornos gráficos y pruebas.
Se referencia en el Capítulo 1, epígrafe 1.5.

C/C++: Potente lenguaje de programación para el desarrollo de aplicaciones.
Se referencia en el Capítulo 1, epígrafe 1.5.

SQL: Acrónimo en ingles de Structured Query Language (Lenguaje de Consulta Estructurado). Es un lenguaje de consulta estructurado de acceso a bases de datos relacionales 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, de una forma sencilla.
Se referencia en el Capítulo 1, epígrafe 1.5.

API: Acrónimo en ingles de Application Programming Interface (Interfaces de Programación de Aplicaciones). Es una interfaz de programación de la aplicación que reune un conjunto de especificaciones de comunicación entre componentes software.
Se referencia en el Capítulo 1, epígrafe 1.6.

Lenguajes Procedurales: Son un lenguajes de programación que están definidos fundamentalmente para la utilización de variables para almacenar valores y para la realización de operaciones con los datos almacenados.
Algunos ejemplos son: FORTRAN, PASCAL y C.
Se referencia en el Capítulo 1, epígrafe 1.5.3.

POSIX: Acrónimo en inglés de Portable Operating System Interface Unix Based (Sistema Operativo Portable Basado en UNIX). Una familia de estándares de llamadas al sistema. Intenta estandarizar las interfaces de los sistemas operativos para que las aplicaciones se ejecuten en distintas plataformas.
Se referencia en el Capítulo 1, epígrafe 1.7.

IDE: Acrónimo en inglés de Integrated Drive Electronics (Entorno Integrado de Desarrollo). Estándar utilizado para crear la conexión de unidades de almacenamiento (discos duros, lectores de CD, etc) a los ordenadores.
Se referencia en el Capítulo 1, epígrafe 1.8.

GNU/GPL: Acrónimo en inglés de General Public License (Licencia Pública General). Es una licencia que está orientada principalmente a proteger la libre distribución, modificación y uso de software. Su propósito es declarar que el software cubierto por esta licencia es software libre y protegerlo de intentos de apropiación que restrinjan esas libertades a los usuarios.
Se referencia en el Capítulo 2, epígrafe 2.6.2.