Universidad de las Ciencias Informáticas

Facultad 1

Sistema informático para la gestión del proceso de matrícula en las asignaturas optativas de la Facultad 1 de la Universidad de las Ciencias Informáticas

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autor:

Roberto Noel Acosta Cabrera

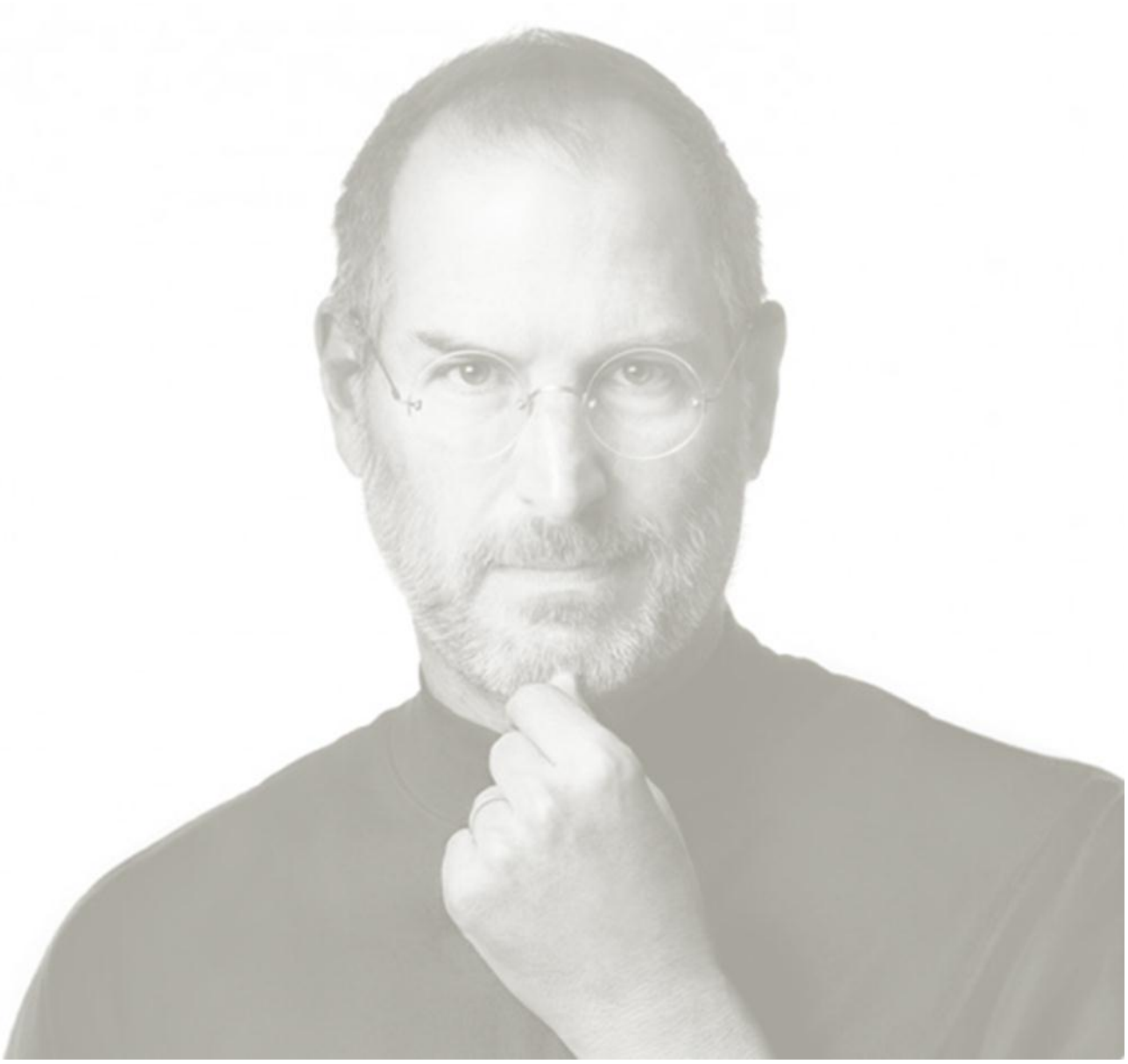
Tutores:

MSc. Eylín Hernández Luque

MSc. Yurisbel Vega Ortiz

La Habana, 17 de Julio del 2015

"Año 57 de la Revolución"



“LA GENTE QUE ESTÁ LO SUFICIENTEMENTE LOCA COMO PARA PENSAR QUE PUEDEN CAMBIAR EL MUNDO, SON LAS QUE LOGRAN HACERLO”

STEVE JOBS

1955-2011

Declaración de autoría

Declaro ser el único autor del trabajo “Sistema informático para la gestión del proceso de matrícula en las asignaturas optativas de la Facultad 1 de la Universidad de las Ciencias Informáticas” y autorizo a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio. Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

_____

Roberto Noel Acosta Cabrera

Firma del Autor

_____

MSc. Yurisbel Vega Ortiz

Firma del Tutor

_____

MSc. Eylín Hernández Luque

Firma del Tutor

A mi maravillosa familia, quienes me han brindado todo lo necesario para poder llegar a donde estoy por haber creído en mí siempre, dándome ejemplo de superación, humildad y sacrificio; enseñándome a valorar todo lo que tengo.

Agradecimientos

- A Dios por haberme otorgado una familia maravillosa, quienes han creído en mí siempre, dándome ejemplo de superación, humildad y sacrificio; enseñándome a valorar todo lo que tengo.

- A Cuba, Fidel y la Revolución por brindarme la oportunidad de formar parte de una patria libre, informatizada y nuestra.

- A mi novia Lidia por formar una parte especial en mi vida, por el cariño, amor, ayuda y comprensión brindada de manera inagotable.

- A todos mis amigos por la ayuda incondicional durante todo este tiempo.

- A todos aquellos que contribuyeron en mi formación académica y profesional: a mis profesores, que compartieron conmigo sus conocimientos a lo largo de mi educación universitaria; especialmente a Serguey González por ser más que mi profesor un gran amigo, a mis tutores de tesis, por su amistad, apoyo y paciencia para la elaboración de este trabajo.

Resumen

Con el avance de las tecnologías se precisa de una gestión de la información de manera digital, que permita una mejor administración y acceso a los datos. En el Vicedecanato de Formación de la Facultad 1 de la Universidad de las Ciencias Informáticas, se lleva a cabo el proceso de gestión de matrícula de las asignaturas optativas. Este proceso, se desarrolla de forma manual, lo que trae consigo una mayor inversión de tiempo, información duplicada y errores en los datos que se adquieren. Para dar solución a las insatisfacciones existentes en la investigación, se propone como objetivo: desarrollar un sistema informático, sustentado en las nuevas tecnologías, para que contribuya al proceso de gestión y almacenamiento de la información de la matrícula de las asignaturas optativas de la Facultad 1 de la Universidad de las Ciencias Informáticas. Para valorar la fundamentación teórica y comprobar la utilización práctica de la propuesta, se aplican métodos teóricos y empíricos, así como diferentes pruebas de software. Para la implementación de la propuesta, se utilizó como framework de desarrollo Symfony y Bootstrap, para el modelado de los artefactos, Visual Paradigm como herramienta CASE y XP como la metodología de desarrollo. El aporte práctico radica en identificar y proponer cómo desarrollar el proceso de matrícula de manera que fortalezca la correcta gestión de la información, la identificación de los recursos necesarios y la minimización de pérdida de tiempo.

Palabras clave: matrícula, asignaturas optativas, gestión de la información, almacenamiento de la información.

INTRODUCCIÓN

En los últimos años las Tecnologías de la Información y Comunicación (TIC) han tenido una vertiginosa evolución. Las mismas, día a día se han incorporado con gran relevancia a todas las esferas de la sociedad principalmente en la educación, transformando el proceso de enseñanza-aprendizaje. La Universidad de las Ciencias Informáticas (UCI) como parte del sistema de Educación Superior Cubano tiene un plan de estudio definido el cual incluye, en su parte flexible, asignaturas optativas.

En consecuencia con el concepto de asignatura optativa planteado en la Resolución 210/07 Reglamento para el Trabajo Docente y Metodológico en la Educación Superior de Cuba, una asignatura optativa incluye un conjunto de conocimientos organizados de forma estructurada de cualquier materia o tema. Estos conocimientos, los reciben los estudiantes de manera optativa u opcional con el objetivo de prepararse profesionalmente, así como desarrollar habilidades y capacidades. Las asignaturas optativas forman parte del plan de estudio del programa académico y contribuyen a la formación del estudiante.

En la UCI, curso tras curso se lleva a cabo el proceso de matrícula de las asignaturas optativas, donde participa el Vicedecanato de Formación de la Facultad 1, los Departamentos Docentes que ofertan las asignaturas optativas y los estudiantes. Fue posible identificar contradicciones en este proceso, a partir de la exploración de la realidad, el análisis documental (la búsqueda bibliográfica y teorías, análisis de criterios de autores, revisión de documentos), las entrevistas, las encuestas y algunas consideraciones personales del investigador.

El proceso de matrícula de las asignaturas optativas, tiene insuficiencias en cuanto a su adecuada administración y gestión, ya que los estudiantes tienen que matricularse personalmente, lo que conlleva a que tengan que moverse obligatoriamente hacia el docente. A la vez, trae consigo que los estudiantes se agrupen en los pasillos frente a los departamentos, lo que obstruye el paso e interrumpe el flujo normal de entrada y salida de las oficinas.

Las convocatorias para matricular en las asignaturas optativas, se publican a principios de los semestres del curso académico y los estudiantes intentan matricular el primer día, para tener la oportunidad de escoger las asignaturas de su interés. Lo que conlleva, a tener concentraciones de estudiantes frente a los departamentos docentes de la facultad 1. Esta acumulación de estudiantes frente a las oficinas afecta el normal funcionamiento de un día de trabajo, ya que provoca ruido y desorden.

La matrícula de los estudiantes en las asignaturas optativas se registra en ficheros *Excel*, lo que provoca que no se pueda comprobar que un estudiante matricule en una asignatura optativa que ya haya vencido o que se imparta en el mismo horario de otra. Esta herramienta que hoy se utiliza, no facilita un reporte que permita tener un control de los estudiantes que faltan por matricular, así como de las cantidades que se han matriculado en cada asignatura optativa por año académico.

Las dificultades anteriormente descritas permiten identificar el siguiente **problema científico**: ¿Cómo perfeccionar el proceso de gestión y almacenamiento de la información de la matrícula de las asignaturas optativas de la Facultad 1 de la Universidad de las Ciencias Informáticas?

El **objeto de estudio** de la investigación, es el proceso de gestión y almacenamiento de la información de la matrícula de las asignaturas optativas.

Se concibe como **objetivo de la investigación**, desarrollar un sistema informático, sustentado en las nuevas tecnologías, para que contribuya al proceso de gestión y almacenamiento de la información de la matrícula de las asignaturas optativas de la Facultad 1 de la Universidad de las Ciencias Informáticas.

Para la organización del trabajo surgen los siguientes **objetivos específicos:**

1. Definir el marco teórico de la investigación.
2. Desarrollar la aplicación web que permita la gestión de los procesos de matrícula de las asignaturas optativas en la Facultad 1.
3. Validar la aplicación web mediante pruebas de *software*.

De la relación entre el problema, el objeto de estudio y el objetivo se identifica como **campo de acción**, los sistemas informáticos para la gestión y almacenamiento de la información de la matrícula de las asignaturas optativas.

El proceso de investigación se orientó a través de las siguientes tareas investigativas:

1. Comprobación de la validez del problema.

2. Sistematización de los referentes teóricos del proceso de gestión y almacenamiento de la información de la matrícula de las asignaturas optativas.

3. Fundamentación de las tecnologías, herramientas, metodología y lenguajes necesarios para implementar la propuesta de solución.

4. Identificación de los requisitos funcionales y no funcionales.

5. Definición de patrones de diseño.

6. Confección de las tarjetas CRC (*Class –Responsability -Colaboration*).

7. Implementación de la propuesta de solución.

9. Validación de la propuesta a través de las pruebas.

Hipótesis: Si se desarrolla una aplicación web para la gestión y almacenamiento de la información de la matrícula de las asignaturas optativas, sustentada en las nuevas tecnologías, se contribuye a la perfección de dicho proceso en la Facultad 1 de la Universidad de las Ciencias Informáticas.

Variable Independiente:

- Aplicación web para la gestión y almacenamiento de la información de la matrícula de las asignaturas optativas.

Variable Dependiente:

- Perfeccionamiento de la gestión y almacenamiento de la información de la matrícula de las asignaturas optativas.

Para definir cómo desarrollar la investigación se confeccionó el diseño metodológico de la investigación.

El tipo de investigación es descriptivo, porque se profundiza en las características del proceso de gestión y almacenamiento de la información de la matrícula de las asignaturas optativas y cómo influye la implementación de un sistema sustentando en las nuevas tecnologías, en la agilización y centralización de la información en la Facultad 1 de la Universidad de las Ciencias Informáticas.

La población definida para el desarrollo de la investigación está comprendida por: el Vicedecano de Formación de la Facultad 1, los Jefes de Departamentos Docentes que ofertan las asignaturas optativas y los estudiantes. Las unidades de estudio las constituyen los profesores y estudiantes que llevan a cabo el proceso de gestión y almacenamiento de la información de la matrícula de las asignaturas optativas de la Facultad 1 de la Universidad de las Ciencias Informáticas.

Se utilizaron los métodos teóricos siguientes:

- **Histórico – lógico:** La utilización de las fuentes de información originales, permitió a través del estudio y la investigación de los referentes teóricos del proceso de gestión y almacenamiento de la información de la matrícula de las asignaturas optativas, el descubrimiento de la lógica, el análisis, la evolución, la conceptualización histórica y las tendencias de desarrollo del objeto de estudio.

- **Modelación:** Permitió realizar las actividades encaminadas a la adaptación o construcción de modelos a través de los cuales se obtuvo una representación de la propuesta de solución y a partir de estos implementar el sistema.

- **Analítico – sintético:** Permitió un estudio, análisis y síntesis del proceso de gestión y almacenamiento de la información de la matrícula de las asignaturas optativas, a través de la fundamentación y los razonamientos, lo cual significa coincidir con planteamientos y/o plantear contradicciones, para tener criterios en la toma de decisiones.

- **Sistémico:** Permitió el análisis, la reproducción e integración de las propiedades del objeto que se investiga en otro similar que se construye, para favorecer la aplicación de un sistema informático de gestión y almacenamiento de la información de la matrícula de las asignaturas optativas, sustentado en las nuevas tecnologías, que contribuya a la agilización y centralización de la información en la Facultad 1 de la Universidad de las Ciencias Informáticas.

- **Hipotético deductivo:** Este método permitió definir el procedimiento o camino a seguir durante la investigación, haciendo de su uso una correcta práctica científica.

Para describir las características y obtener la información necesaria del objeto de estudio se utilizaron métodos empíricos, como los siguientes:

- **Entrevistas:** Se realizó para identificar de manera detallada el proceso de gestión y almacenamiento de la información de la matrícula de las asignaturas optativas en la Facultad 1 de la Universidad de las Ciencias Informáticas así como para la captura de requisitos.

- **Análisis documental:** Se utilizó para analizar, clasificar, verificar, seleccionar los contenidos en la bibliografía e informes de investigación, referentes al proceso de gestión y almacenamiento de la información de la matrícula de las asignaturas optativas en la Facultad 1 de la Universidad de las Ciencias Informáticas, por lo que se recurrió a la revisión de documentos, la búsqueda bibliográfica y teorías así como el análisis de criterios de autores.

- **Encuestas:** Fue una fuente para la caracterización inicial del objeto que se investiga; así como para la determinación de las contradicciones y específicamente del problema y el objetivo planteado. Permitió obtener información que caracteriza el estado de opinión de la muestra seleccionada acerca de los inconvenientes e insatisfacciones que se generan durante la ejecución del proceso de matrícula.

- **Criterio de expertos:** Se utilizó para evaluar la propuesta que se presenta en la investigación, permitió la valoración de la actualidad, significación de los resultados y aplicabilidad.

El documento está estructurado en tres capítulos los cuales se describen a continuación:

- **Capítulo 1. Fundamentación teórica:** En este capítulo se estudian los conceptos fundamentales asociados al dominio del problema para un mejor entendimiento de la investigación. Incluye un estado del arte del tema tratado, a nivel internacional y nacional. Así como la descripción de las metodologías, tecnologías y herramientas empleadas en el desarrollo del sistema.
- **Capítulo 2. Análisis y diseño del sistema**: En este capítulo se describen las características de la aplicación a desarrollar. Se realizan las especificaciones de las funcionalidades del sistema y se elabora el plan de entregas y de iteraciones.
- **Capítulo 3. Implementación y pruebas**: En este capítulo se plantea la construcción de la solución propuesta en el Capítulo 2. Se presentan las principales tareas de implementación a realizar y los casos de prueba para las historias de usuario.
- Posteriormente se muestran las **conclusiones**, **recomendaciones** y las referencias **bibliográficas** de la investigación; así como los **anexos** que complementan el contenido esencial del documento escrito.

CAPÍTULO 1. FUNDAMENTACIÓN DE LOS REFERENTES TEÓRICO-METODOLÓGICOS

1.1. Introducción

Con el objetivo de lograr una mayor comprensión del alcance de la investigación y esclarecer el objeto de estudio, se hace una investigación acerca de los conceptos principales de este proceso. Además, se expone una valoración del estado del arte y se analizan las tendencias y tecnologías actuales referentes al objeto y campo de la investigación. En este capítulo, se fundamenta el uso de las herramientas, lenguajes y metodología a utilizar, lo que constituye el basamento teórico del Sistema de Gestión de Asignaturas Optativas.

1.2. Conceptos asociados al objeto de estudio

A continuación se abordarán los principales conceptos asociados al dominio del problema, haciendo énfasis en los conceptos de gestión, matrícula, asignaturas optativas entre otros que propicien un mayor entendimiento del objeto de estudio.

Asignatura

Una asignatura es una materia que se enseña en la escuela, en una universidad o cualquier otro establecimiento educativo y que forma parte de una carrera o curso (1).

Asignaturas optativas

Las asignaturas optativas son aquellas materias que se incluyen en el plan de estudio y de entre las cuales el estudiante selecciona una cantidad determinada para cursar en forma obligatoria. Los contenidos de estas asignaturas tienen como propósito ampliar y actualizar a los estudiantes sobre temas científicos relacionados con la profesión (2). En la UCI, este concepto se adapta a las necesidades y objetivos de la universidad, de tal forma que las asignaturas optativas están incluidas dentro del plan de estudio, cada estudiante debe haber cursado y aprobado un total de 8 asignaturas optativas durante toda la carrera: 2 en el 2do semestre de 3er año, 2 en cada semestre de 4to año y 2 en el primer semestre de 5to.

Matrícula

La matrícula es la inscripción oficial en los registros del Centro de Educación Superior, mediante la cual una persona formaliza o ratifica al inicio de cada período académico su condición de estudiante. Se concede a los ciudadanos que cumplan los requisitos y las normas establecidas en las disposiciones que al efecto dicte el Ministerio de Educación Superior o el organismo especializado al cual se encuentra adscripta la carrera.

Para los estudiantes que cursan estudios en el curso diurno y en el curso para trabajadores, la matrícula implica la obligación de cursar todas las asignaturas establecidas para el año académico que le corresponda en el plan de estudio, que incluyen las asignaturas electivas y optativas que seleccione (4).

Gestión

Se denomina gestión al correcto manejo de los recursos de los que dispone una determinada organización**,** como por ejemplo, empresas, organismos públicos, organismos no gubernamentales (6). En esta investigación se hace referencia a este concepto relacionándolo con el manejo de los datos para realizar una matrícula de una asignatura optativa.

Sistema de Gestión de Matrícula

Se denomina Sistema de Gestión de Matrícula al registro de matrícula estudiantil y a la promoción automática de alumnos al siguiente período lectivo de asignaturas según criterios (7). Se hace referencia a este concepto teniendo en cuenta que se pretende realizar una aplicación web que gestione de manera automática la información relacionada con el proceso de matrícula de las asignaturas optativas.

1.3. Análisis de soluciones existentes

En la actualidad existen mecanismos y *softwares* que posibilitan gestionar la información docente y controlar las asignaturas que conforman el plan de estudio de cualquier carrera o asignaturas optativas que se brindan u ofertan a estudiantes para permitir su capacitación. Los sistemas de gestión de información académica son muy utilizados en cualquier centro de enseñanza gracias a las comodidades que ellos proporcionan al claustro de profesores y a los estudiantes.

1.3.1. Soluciones existentes a nivel internacional

web.unican.es

Es un sistema informático perteneciente a la Universidad Cantabria que se encarga de gestionar Materias, Grupos, Docentes, Centros, Estudiantes, Horarios y otros aspectos que son de interés para la organización (8). Este sistema cuenta con una serie de módulos que permiten gestionar el trámite docente, uno de ellos es la gestión de asignaturas, en el cual publican las asignaturas a cursar junto con los datos pertenecientes a la misma ya sea disciplina o documento asociado. Estas asignaturas son solicitadas por los estudiantes pertenecientes a la institución para luego poder cursarlas.

DocCF, *Software* de Gestión Escolar

DocCF es una aplicación desarrollada por el grupo CF *Developer* para automatizar los procedimientos administrativos, académicos y comerciales de las Instituciones Educativas. El objetivo de este *software* como herramienta, es gestionar los procesos internos y facilitar la coordinación y comunicación entre padres, alumnos, docentes y cargos directivos para ofrecer información estadística sobre dichos procesos y facilitar la toma de decisiones en el proceso de gestión de la institución. Todos los procedimientos están organizados por módulos lo que permite realizar una gestión ordenada y eficiente de la Institución Educativa. De esta manera se puede generar información rápida y completa para el uso o análisis entre todos los componentes de la Comunidad Educativa: cargos directivos, docentes y alumnos (9).

1.3.2. Soluciones existentes a nivel nacional

Sistema Automatizado de Gestión de la Maestría Informática en Salud

Sistema para el control automatizado de la gestión de la Maestría Informática en Salud. El sistema permitirá coordinar actividades docentes así como gestionar la información de estudiantes, profesores y graduados en aras de agilizar los procesos y obtener, de manera oportuna, la información necesaria a cada uno de ellos y a los directivos, para hacer un uso más eficiente y eficaz de los recursos (10).

1.3.3. Soluciones existentes en la UCI

Sistema de Gestión Universitaria

Actualmente, la Universidad de las Ciencias Informáticas, cuenta con un sistema que gestiona las actividades de formación. Este sistema, denominado Gestión Universitaria, está integrado por los siguientes módulos: Pregrado, Posgrado, Producción, Investigación, Ingreso, Ubicación Laboral, Laboratorio, Residencia, Extensión Universitaria, Cooperación Internacional, Biblioteca y Teleformación (11). Este sistema de gestión es uno de los sistemas más usados de la Universidad de las Ciencias Informáticas, pero no cuenta con un módulo que gestione el proceso de matrícula de las asignaturas optativas de la universidad.

Resumen del análisis de sistemas homólogos

El análisis de los sistemas antes descritos sirvieron de referencia para identificar buenas prácticas que pudieran ser incorporadas en la propuesta de solución, ya que los mismos cuentan con diferentes características en cuanto a diseño, pero su objetivo final es la gestión de datos relacionadas con la educación.

En el caso de web.unican.es cuenta con un diseño acogedor y realiza una correcta manipulación de la información, pero el mismo tiene como objetivo satisfacer solamente las necesidades internas de la institución. DocCF es una aplicación de escritorio que gestiona la información relacionada con las actividades docentes del mismo, se tomará como ejemplo la estructura en que muestra la información relacionada con los estudiantes. El Sistema Automatizado de Gestión de la Maestría Informática en Salud en estos momentos se encuentra en desarrollo pero la documentación del mismo sirvió como guía para la investigación. Como último Gestión Universitaria fue uno de los sistemas que brindó un gran aporte a la investigación, ya que algunos de sus servicios serán empleados en la propuesta de solución. Estos sistemas no se tomaron como una solución al problema en cuestión, pues no cuentan con las características necesarias que posibiliten la gestión de matrícula de asignaturas optativas.

1.4. Caracterización de los sistemas informáticos

Aplicación de escritorio

Las aplicaciones de escritorio son programas que se instalan en el ordenador y sirven para realizar diferentes tareas como gestión de pedidos, gestión de incidencias, contabilidad, almacenamiento de datos, comunicación interna y externa, gestión de personal, gestión de empresas, entre otras.

Las aplicaciones pueden trabajar sobre diferentes ámbitos y plataformas. El ámbito de trabajo puede ser (12):

- Local solo en el ordenador instalado.
- Red interna entre diversos ordenadores de la red o con un servidor.
- Internet sobre una base de datos externa.

La principal **ventaja** es la rapidez de uso ya que puede incorporar todos los controles de escritorio y todos los eventos asociados a ellos.
Como principal **desventaja** se encuentra la gestión de actualizaciones que obliga a actualizar todos los programas instalados en cada puesto de la empresa cuando se implemente evoluciones o se corrijan fallos(13).

Las aplicaciones de escritorio no resultan una solución eficiente para el problema que se presenta, pues agrupa la gestión de la información a una o solo algunas estaciones de trabajo donde se hace necesario instalar todas las herramientas que requiere el *software* para su funcionamiento, limitando su uso solo a aquellos usuarios que tengan acceso a dichas computadoras.

Aplicaciones web

Las aplicaciones web pueden realizar las mismas funciones que las señaladas en las aplicaciones de escritorio, pero son accesibles a través de cualquier navegador por Internet, por lo que no tienen que ser instaladas en ningún ordenador. Esto facilita su trabajo multiplataforma y su acceso distribuido de la información.

Una de las grandes ventajas es que no es necesario invertir en grandes máquinas aunque la aplicación sea muy potente, ya que puede estar instalada en servidores en la nube.

Ventajas de las aplicaciones web

- Las aplicaciones web se integran fácilmente en otros procedimientos web del lado del servidor, tales como el correo electrónico y la búsqueda.
- Proporcionan compatibilidad entre plataformas en la mayoría de los casos (Windows, Mac, Linux) debido a que operan dentro de una ventana del navegador web (14).

Como principal **desventaja** se encuentra que las aplicaciones web se basan en archivos de la aplicación que acceden a servidores remotos a través de Internet. Por lo tanto, cuando la conexión se interrumpe, la aplicación queda fuera de servicio hasta que se restablezca la misma (15).

En esta investigación se determina que una aplicación web constituye el sistema informático adecuado para llevar a cabo el proceso de matrícula en asignaturas optativas de la Facultad 1 de la Universidad de las Ciencias Informáticas ya que la misma cuenta con la característica de permitir el acceso al sistema desde la residencia u otro lugar de la universidad.

1.5. Tecnologías a utilizar

1.5.1. Servidor web

Un servidor web es un programa informático que procesa una aplicación del lado del servidor realizando conexiones bidireccionales y/o unidireccionales y síncronas o asíncronas con el cliente generando o cediendo una respuesta en cualquier lenguaje o aplicación del lado del cliente (16).

Entre los tipos más encontrados de servidores web podemos encontrar MICROSOFT IIS, SUN JAVA SYSTEM WEB SERVER, WAMP, XAMPP, APPSERV, APACHE. Se seleccionó este último en su versión 2.2.22 para el desarrollo del sistema propuesto por las características descritas a continuación(17):

- Es multiplataforma, lo que lo hace prácticamente universal. Apache es un servidor altamente configurable de diseño modular. Es muy sencillo ampliar las capacidades del servidor web Apache.
- Apache es una tecnología gratuita de código fuente abierto.

- Permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor. Es posible configurar Apache para que ejecute un determinado script cuando ocurra un error en concreto.

1.5.2. Lenguajes de programación

Hypertext Preprocessor o Preprocesador de Hipertexto (PHP)

Lenguaje de programación usado generalmente en la creación de contenidos para sitios web. Es un lenguaje interpretado especialmente usado para crear contenido dinámico web y aplicaciones para servidores. Generalmente los *scripts* en PHP se embeben en otros códigos como HTML, ampliando las posibilidades del diseñador de páginas web enormemente. La interpretación y ejecución de los *scripts* PHP se hacen en el servidor, el cliente (un navegador que pide una página web) solo recibe el resultado de la ejecución (18).

Este lenguaje es gratuito y multiplataforma. Rápido, con una gran librería de funciones y mucha documentación. Este lenguaje de programación está preparado para realizar diferentes tipos de aplicaciones web gracias a la extensa librería de funciones con la que está dotado.

PHP cuenta con diferentes funcionalidades, una de las más importantes es la **compatibilidad con las bases de datos** más comunes, como PostgreSQL. Incluyendo funciones para el **envío de correo electrónico** y ***upload* de archivos** (19).

En la Universidad de Ciencias Informáticas existe una Comunidad de Desarrollo de PHP la cual sirve de apoyo para la implementación del sistema informático que se desea desarrollar.

Para el desarrollo de la propuesta de solución se empleará el lenguaje de programación PHP en su versión 5.3.6 por las razones antes expuestas.

***HyperText Markup Language* o Lenguaje de Marcas de Hipertexto (HTML)**

HTML es el lenguaje con el que se escribe la estructura y la semántica del contenido de un documento web (20). Se encarga de desarrollar una descripción sobre los contenidos que aparecen como textos y sobre su estructura, complementando dicho texto con diversos objetos como fotografías, animaciones. Es un lenguaje muy simple y general que sirve para definir otros lenguajes que tienen que ver con el formato de los documentos (21). Para el desarrollo del sistema se empleará el lenguaje HTML en su versión 5.

Hojas de Estilo en Cascada - *Cascading Style Sheets* (CCS)

CSS es un lenguaje que describe la presentación de los documentos estructurados en hojas de estilo para diferentes métodos de interpretación, es decir, describe cómo se va a mostrar un documento en pantalla.

CSS 3 es una especificación desarrollada por el W3C (*World Wide Web Consortium*) para permitir la separación de los contenidos de los documentos escritos en HTML, XML, XHTML, SVG, o XUL de la presentación del documento con las hojas de estilo, incluyendo elementos tales como los colores, fondos, márgenes, bordes, tipos de letra entre otros, modificando la apariencia de una página web de una forma más sencilla, permitiendo a los desarrolladores controlar el estilo (22).

JavaScript

JavaScript es un lenguaje con muchas posibilidades, utilizado para crear pequeños programas que luego son insertados en una página web y en programas más grandes orientados a objetos mucho más complejos. Con *JavaScript* se pueden crear diferentes efectos e interactuar con los usuarios. No requiere de compilación ya que el lenguaje funciona del lado del cliente, los navegadores son los encargados de interpretar estos códigos. *Java* por su parte tiene como principal característica ser un lenguaje independiente de la plataforma. Se puede crear todo tipo de programa que puede ser ejecutado en cualquier ordenador del mercado: Linux, Windows, Apple (23). En el sistema se utilizará *Java* Script en su versión 1.8.

1.5.3. Gestor de Base de Datos PostgreSQL

PostgreSQL es un potente sistema de base de datos objeto-relacional de código abierto. Cuenta con más de 15 años de desarrollo activo y una arquitectura probada que se ha ganado una sólida reputación de fiabilidad e integridad de datos. Se ejecuta en los principales sistemas operativos que existen en la actualidad como:

- Linux.
- UNIX (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64).
- Windows.

PostgreSQL tiene soporte completo para claves foráneas, uniones, vistas, disparadores y procedimientos almacenados (en varios lenguajes). Incluye la mayoría de los tipos de datos del SQL 2008, incluyendo INTEGER, NUMÉRICO, BOOLEAN, CHAR, VARCHAR, DATE, INTERVAL, Y TIMESTAMP. Cuenta con interfaces nativas de programación para C / C + +, *Java*, Perl, Python, Ruby, entre otros, y la documentación que actualmente existente en la Universidad de las Ciencias Informáticas es realmente excepcional, pues la misma cuenta con una Comunidad de Desarrolladores para PostgreSQL y en el trayecto de la carrera se imparten asignaturas que incluyen el trabajo con este gestor de bases de datos, lo que facilita el desempeño con el mismo. En el presente trabajo se utilizará PostgreSQL en la versión 9.4 (24).

1.5.4. Administrador de Base de Datos pgAdmin

pgAdmin es una aplicación gráfica para gestionar el gestor de bases de datos PostgreSQL. Está escrita en C++ usando la librería gráfica multiplataforma wxWidgets, lo que permite que se pueda usar en Linux, FreeBSD, Solaris, Mac OS X y Windows. Es capaz de gestionar versiones a partir de PostgreSQL 7.3 ejecutándose en cualquier plataforma, así como versiones comerciales de PostgreSQL como *Pervasive Postgres, EnterpriseDB, Mammoth Replicator y SRA PowerGres* (25).

pgAdmin está diseñada para responder a las necesidades de todos los usuarios, desde escribir consultas SQL simples hasta desarrollar bases de datos complejas. El interfaz gráfico soporta todas las características de PostgreSQL y facilita enormemente la administración (25). Por lo anteriormente planteado se utilizará este administrador de bases de datos en la versión 1.8.

1.5.5. Metodologías de desarrollo de software

Un proceso de *software* detallado y completo suele denominarse "Metodología". Las metodologías se basan en una combinación de los modelos de proceso genéricos (cascada, evolutivo, incremental, espiral entre otros). Adicionalmente una metodología debería definir con precisión los artefactos, roles y actividades involucrados, junto con prácticas y técnicas recomendadas, guías de adaptación de la metodología al proyecto, guías para uso de herramientas de apoyo. Habitualmente se utiliza el término "método" para referirse a técnicas, notaciones y guías asociadas, que son aplicables a una (o algunas) actividades del proceso de desarrollo, por ejemplo, suele hablarse de métodos de análisis y/o diseño (26).

1.5.6. Metodología XP (Extreme Programing o Programación Extrema*)*

Es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de *software*, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores y propiciando un buen clima de trabajo. XP se basa en la retroalimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico(27).

El ciclo de vida ideal de XP consiste de seis fases (28):

I. Exploración

   En esta fase los clientes plantean a grandes rasgos las historias de usuario que son de interés para la primera entrega del producto. Al mismo tiempo el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto. Se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo.

II. Planificación de la Entrega (*Release*)

En esta fase el cliente establece la prioridad de cada historia de usuario y correspondientemente, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente. Una entrega debería obtenerse en no más de tres meses. La planificación se puede realizar basándose en el tiempo o el alcance.

III. Iteraciones

Esta fase incluye varias iteraciones sobre el sistema antes de ser entregado. El Plan de Entrega está compuesto por iteraciones de no más de tres semanas. En la primera iteración se puede intentar establecer una arquitectura del sistema que pueda ser utilizada durante el resto del proyecto. Los elementos que deben tomarse en cuenta durante la elaboración del Plan de la Iteración son: historias de usuario no abordadas, velocidad del proyecto, pruebas de aceptación no superadas en la iteración anterior y tareas no terminadas en la iteración anterior.

IV. Producción

La fase de producción requiere de pruebas adicionales y revisiones de rendimiento antes de que el sistema sea trasladado al entorno del cliente. Al mismo tiempo, se deben tomar decisiones sobre la inclusión de nuevas características a la versión actual, debido a cambios durante esta fase. Es posible que se rebaje el tiempo que toma cada iteración, de tres a una semana. Las ideas que han sido propuestas y las sugerencias son documentadas para su posterior implementación.

V. Mantenimiento

Mientras la primera versión se encuentra en producción, el proyecto XP debe mantener el sistema en funcionamiento al mismo tiempo que desarrolla nuevas iteraciones. Para realizar esto se requiere de tareas de soporte para el cliente. De esta forma, la velocidad de desarrollo puede bajar después de la puesta del sistema en producción. La fase de mantenimiento puede requerir nuevo personal dentro del equipo y cambios en su estructura.

VI. Muerte del Proyecto

Es cuando el cliente no tiene más historias para ser incluidas en el sistema. Esto requiere que se satisfagan las necesidades del cliente en otros aspectos como rendimiento y confiabilidad del sistema. Se genera la documentación final del sistema y no se realizan más cambios en la arquitectura. La muerte del proyecto también ocurre cuando el sistema no genera los beneficios esperados por el cliente o cuando no hay presupuesto para mantenerlo.

Para llevar a cabo la realización de este trabajo se determina usar la Metodología XP teniendo en cuenta que es un proyecto pequeño con relativa simplicidad en las implementaciones, con requisitos cambiantes y el cliente mantiene constante comunicación con el desarrollador.

1.5.7. Lenguaje Unificado de Modelado (UML)

UML (*Unified Modeling Language*) o Lenguaje Unificado de Modelado prescribe un conjunto de notaciones y diagramas estándar para modelar sistemas orientados a objetos y describe la semántica esencial de estos diagramas y los símbolos en ellos utilizados(3).

UML ofrece 9 tipos de diagramas con los cuales se pueden modelar sistemas:

- Diagrama de Casos para Uso para modelar los procesos "*business".*
- Diagrama de Secuencia para modelar el paso de mensajes entre objetos.
- Diagrama de Colaboración para modelar interacciones entre objetos.
- Diagrama de Estado para modelar el comportamiento de los objetos en el sistema.
- Diagramas de Actividad para modelar el comportamiento de los Casos de Uso, objetos u operaciones.
- Diagrama de Clases para modelar la estructura estática de las clases en el sistema.
- Diagrama de Objetos para modelar la estructura estática de los objetos en el sistema.
- Diagramas de Componentes para modelar componentes.
- Diagrama de Implementación para modelar la distribución del sistema.

UML no es, por tanto, un método, sino varios. Se trata de una estandarización o consolidación de muchas notaciones y modelos usados anteriormente. Por las características mencionadas se escoge UML en su versión 2.0 como lenguaje de modelado.

1.5.8. Herramienta CASE (Computer Aided Software Engineering o Ingeniería de *Software* Asistida por Ordenador*)*

El propósito de una herramienta CASE es dar soporte automatizado para la aplicación de todas o algunas técnicas usadas por una o varias metodologías (29):

Las metodologías incluyen gran cantidad de técnicas y el esfuerzo de documentación (y actualización de dicha documentación) es por lo general considerable. Por lo tanto, es difícil aplicar una metodología sin la ayuda de una herramienta CASE. Así, los beneficios de utilización de un CASE se entremezclan con los beneficios de aplicar una metodología con éxito. El valor agregado indudable de utilizar un CASE es el aumento en la productividad en las actividades soportadas por la herramienta.

El uso de metodologías de desarrollo junto a herramientas CASE sin lugar a dudas ofrece la mejor alternativa actual para enfrentar proyectos de desarrollo de *software* de complejidad y/o envergadura. El énfasis en planificación, análisis y diseño promovido por una herramienta CASE tiene un fuerte impacto y recompensa en la mejora de la calidad del producto obtenido y en el aumento de productividad (disminución de tiempos, costes y esfuerzos) en las actividades de desarrollo y mantenimiento. El beneficio adicional obtenido por la utilización de un CASE actual (si se compara con la utilización de una metodología sin el uso de un CASE) se representa en los siguientes aspectos:

- Facilita la verificación y mantenimiento de la consistencia de la información del proyecto.
- Facilita el establecimiento de estándares en los procesos de desarrollo y documentación.
- Facilita el mantenimiento del sistema y las actualizaciones de su documentación.
- Facilita la aplicación de las técnicas de una metodología.

Para el desarrollo del presente trabajo se selecciona la herramienta CASE Visual Paradigm.

Visual Paradigm

Visual Paradigm para UML es una herramienta para desarrollo de aplicaciones utilizando modelado UML ideal para Ingenieros de *Software*, Analistas de Sistemas y Arquitectos de Sistemas que están interesados en construcción de sistemas a gran escala y necesitan confiabilidad y estabilidad en el desarrollo orientado a objetos. Para el desarrollo del trabajo en cuestión se utilizará *Visual Paradigm* en su versión 8.0.

Visual Paradigm también ofrece(30):

- Navegación intuitiva entre la escritura del código y su visualización.
- Potente generador de informes en formato PDF/HTML.
- Documentación automática Ad-hoc.
- Ambiente visual superior de modelado.
- Sofisticado diagramador automáticamente de *layout.*

1.5.9. Entorno de Desarrollo Integrado (IDE)

NetBeans

Netbeans es un entorno de desarrollo gratuito y de código abierto, permite el uso de un amplio rango de tecnologías de desarrollo tanto para escritorio, como aplicaciones web, o para dispositivos móviles. Da soporte a las siguientes tecnologías: *Java*, PHP, Groovy, C/C++, HTML5. Además puede instalarse en varios sistemas operativos: Windows, Linux, Mac OS, presenta aplicaciones para la detección y tratamiento de errores, también cuenta con una amplia comunidad de desarrollo y soporte (31). Se utilizará en la versión 7.4.

1.5.10. Framework de desarrollo

El concepto *framework* se emplea en muchos ámbitos del desarrollo de *software*, no solo en el ámbito de aplicaciones web. Se puede encontrar *framework* para el desarrollo de aplicaciones médicas, de visión por computador, para el desarrollo de juegos, entre otros. En general, con el término framework, se hace referencia a una estructura de *software* compuesta de componentes personalizables e intercambiables para el desarrollo de una aplicación.

Un *framework* se puede considerar como una aplicación genérica incompleta y configurable a la que se puede añadir las últimas piezas para construir una aplicación concreta. Los objetivos principales que persigue un *framework* son: acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones. Un *framework* web, por tanto, se puede definir como un conjunto de componentes (por ejemplo clases en *java* y descriptores y archivos de configuración en XML) que componen un diseño reutilizable que facilita y agiliza el desarrollo de sistemas web (32).

Symfony

Symfony es un completo *framework* diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones web. Separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación.

Symfony está desarrollado completamente con PHP y es compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft. Se puede ejecutar tanto en plataformas *nix (Unix, Linux, entre otras.) como en plataformas Windows (33). Symfony se utilizará en su versión 2.6.

Bootstrap

Es un *framework* originalmente creado por Twitter, que permite crear interfaces web con CSS y *Java*Script, cuya particularidad es la de adaptar la interfaz del sitio web al tamaño del dispositivo en que se visualice. Es decir, el sitio web se adapta automáticamente al tamaño de una PC, una Tablet u otro dispositivo. Esta técnica de diseño y desarrollo se conoce como "*responsive design*" o diseño adaptativo.

El beneficio de usar *responsive design* en un sitio web, es principalmente que el sitio web se adapta automáticamente al dispositivo desde donde se acceda. Lo que se usa con más frecuencia es un módulo de CSS3 que permite la representación de contenido para adaptarse a condiciones como la resolución de la pantalla y permitiendo editar las dimensiones de contenido en porcentajes, permite también tener una Web muy fluida capaz de adaptarse a casi cualquier tamaño de forma automática.(34) Bootstrap se utilizará en su versión 3.

1.6. Conclusiones

En este capítulo se realizó una sistematización de los fundamentos teórico-metodológicos del proceso de gestión y almacenamiento de la información de la matrícula de las asignaturas optativas, lo que permitió concluir lo siguiente:

- En la facultad 1 de la UCI, se desarrolla en cada curso académico la actividad de matricular a los estudiantes en las asignaturas optativas, compréndase que las asignaturas optativas son un conjunto de conocimientos organizados de forma estructurada de cualquier materia o tema. Estos conocimientos, los reciben los estudiantes de manera optativa u opcional con el objetivo de prepararse profesionalmente, así como desarrollar habilidades y capacidades. Las asignaturas optativas forman parte del plan de estudio del programa académico y contribuyen a la formación del estudiante.
- El análisis de los sistemas de homólogos, permitió identificar que aunque no cuentan con las características necesarias que posibiliten la gestión de matrícula de asignaturas optativas, se pueden identificar buenas prácticas para el desarrollo de una nueva propuesta.
- La fundamentación de las tendencias y tecnologías actuales referentes al objeto y campo de la investigación, permitió identificar las herramientas, lenguajes y metodología a utilizar.

.

CAPÍTULO 2. PROPUESTA DE SOLUCIÓN

2.1. Introducción

Con el objetivo de favorecer una mejor comprensión de la situación problemática presentada en la introducción, se realiza un análisis, que está en correspondencia con las conclusiones del capítulo uno, que fundamenta la solución propuesta. En este capítulo se diseña la propuesta del Sistema de Gestión de Asignaturas Optativas para la facultad 1, de modo que se representa el contexto en que se emplaza, definiéndose conceptos que se agrupan en un Modelo de Dominio para identificar correctamente los requisitos y poder construir un sistema consistente. Además, se realiza una descripción general de los requisitos funcionales y no funcionales del sistema a desarrollar. Y se incluyen las descripciones de los actores, historia de usuario y las tarjetas CRC.

2.2. Modelo de Dominio (MD)

XP no cuenta con un flujo de trabajo para definir el negocio, pero posibilita la especificación de este mediante la forma más cómoda y entendible para aquellas personas que trabajan en el desarrollo de la solución; siempre que no propicie demoras en la entrega del producto final. La realización del modelo de dominio fue posible gracias a los resultados obtenidos durante las entrevistas realizadas al cliente, la aplicación de este método permitió tener una mayor comprensión de los objetos del dominio y sus relaciones.

Un Modelo de Dominio es un artefacto de la disciplina de análisis, construido con las reglas de UML durante la fase de concepción, presentado como uno o más diagramas de clases que contiene, no conceptos propios de un sistema de *software* sino de la propia realidad física.

Los modelos de dominio pueden utilizarse para capturar y expresar el entendimiento ganado en un área bajo análisis como paso previo al diseño de un sistema, ya sea de *software* o de otro tipo. Similares a los mapas mentales utilizados en el aprendizaje (35).

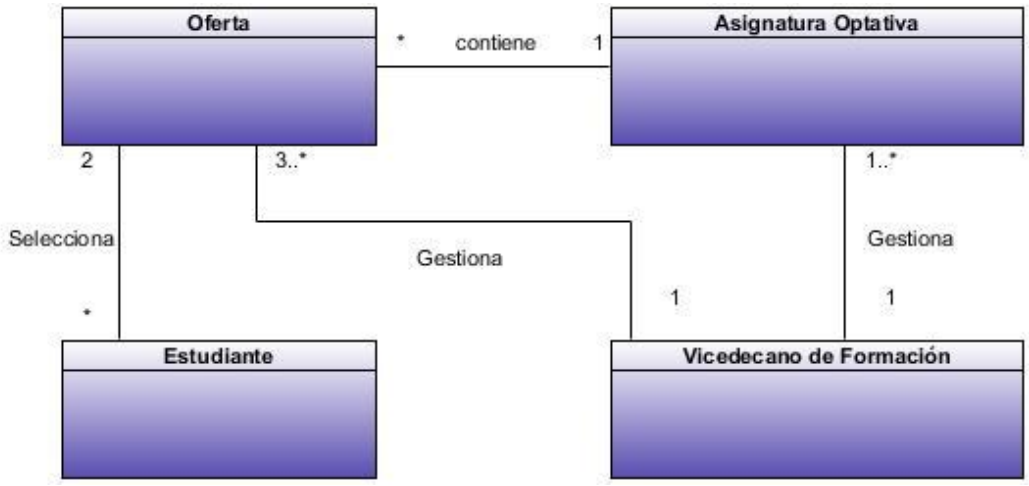


Figura 1 Modelo de Dominio

2.3. Descripción de los objetos del dominio

Vicedecano de Formación: Es el encargado de administrar las asignaturas optativas así como confeccionar las ofertas que se le brindan a los estudiantes.

Estudiante: Encargado de seleccionar las ofertas de asignaturas optativas a matricular que le proporciona el Vicedecano de Formación.

Asignatura Optativa: Son las asignaturas que pertenecen a la parte flexible del plan de formación definido en la UCI, en este caso son definidas en el Vicedecanato de Formación para incluirse en las ofertas que se le brindan a los estudiantes.

Oferta: Son las opciones que el Vicedecano de Formación le brinda a los estudiantes para que seleccionen en cuál asignatura optativa matricular.

2.4. Fase de exploración

La fase de exploración es la primera fase de la metodología XP, en la cual, los clientes plantean a grandes rasgos las funcionalidades que son de interés para la elaboración del producto, transformándose las mismas en historias de usuario.

Partiendo de la información obtenida, el equipo de desarrollo evaluará de forma general el tiempo de desarrollo, se familiarizará con las herramientas, tecnologías y prácticas que serán utilizadas en el proyecto, además, se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo para ello.

La fase de exploración toma de pocas semanas a pocos meses, dependiendo de la habilidad que tengan los programadores con las tecnologías seleccionadas. Las estimaciones realizadas en esta etapa son primarias, pues estas se basan en datos de alto nivel los cuales pueden variar a medida que se analicen con mayor cuidado y detalle en las siguientes fases (36).

2.4.1. Personal relacionado con el sistema

Uno de los principales factores que se deben tener en cuenta cuando se comienza el desarrollo de un sistema informático es la delimitación de la audiencia a la cual va dirigido el mismo, lo cual en algunas ocasiones, puede referirse a un personal relacionado con el sistema o alguien completamente ajeno a él. Se debe especificar que esta audiencia a su vez puede ser dividida en grupos atendiendo a sus necesidades. Se define como persona relacionada con el sistema, a toda aquella que de una manera u otra interactúe con este, obteniendo un resultado de uno o varios procesos que se ejecutan en el mismo. También son considerados como personal relacionado con el sistema, aquellos que se encuentran involucradas en dichos procesos, participen en ellos, pero no obtienen ningún resultado de valor. A continuación se muestra una tabla que muestra la relación entre los actores y la aplicación web.

Tabla 1 Personal relacionado con el sistema

Personal	Descripción
Vicedecano de Formación de la Facultad 1	Tiene completo acceso al sistema y posee el rol de Administrador.
Estudiantes de la Facultad 1	Tienen solamente acceso al módulo de estudiantes, poseen el rol de Estudiante.

A continuación se describe la lista de reserva del producto, compuesta por los requisitos funcionales y los requisitos no funcionales de la aplicación web. Los requisitos funcionales fueron definidos durante las entrevistas realizadas al cliente.

2.4.2. Requisitos funcionales de la aplicación web

Los requerimientos funcionales son declaraciones de los servicios que proveerá el sistema, de la manera en que este reaccionará a entradas particulares (38).

RF_1. Autenticar usuario.

RF_2. Mostar la información de usuario.

RF_3. Realizar solicitud.

3.1 Aceptar la solicitud de matrícula.

3.2 Mostrar el pre-listado de las matrículas en las asignaturas optativas.

3.3 Mostrar el programa (P1) de la asignatura optativa.

RF_4. Gestionar las solicitudes realizadas.

4.1 Cancelar la solicitud de la matrícula en las asignaturas optativas.

4.2 Mostrar el programa (P1) de la asignatura optativa.

RF_5. Mostrar asignaturas cursadas.

RF_6. Gestionar facultades.

6.1 Mostrar las facultades.

6.2 Modificar la información de las facultades.

RF_7. Gestionar período lectivo.

7.1 Actualizar el período.

7.2 Mostrar el período.

RF_8. Gestionar turnos docentes.

8.1 Insertar turno docente.

8.2 Mostrar turnos docentes.

8.3 Eliminar turno docente.

8.4 Actualizar turno docente.

RF_9. Gestionar composición.

9.1 Actualizar la cantidad de asignaturas optativas a cursar por semestre.

9.2 Eliminar la cantidad de asignaturas optativas a cursar por semestre.

9.3 Mostrar la cantidad de asignaturas optativas a cursar por semestre.

9.4 Insertar la cantidad de asignaturas optativas a cursar por semestre.

RF_10. Gestionar años académicos.

10.1 Actualizar los años académicos.

10.2 Eliminar los años académicos.

10.3 Mostrar los años académicos.

10.4 Insertar los años académicos.

RF_11. Gestionar disciplinas.

11.1 Eliminar las disciplinas.

11.2 Mostrar las disciplinas.

11.3 Insertar las disciplinas.

11.4 Actualizar las disciplinas.

RF_12. Gestionar Asignaturas.

12.1 Actualizar las asignaturas.

12.2 Eliminar las asignaturas.

12.3 Mostrar las asignaturas.

12.4 Insertar las asignaturas.

RF_13. Cargar datos.

13.1 Cargar Datos.

13.2 Eliminar Datos.

RF_14. Generar reportes.

14.1 Generar el reporte de los estudiantes por grupo.

14.2 Generar el reporte de las asignaturas por año.

14.3 Generar el reporte de los estudiantes por cantidad de matrículas.

14.4 Generar el reporte total de las asignaturas.

14.5 Generar el reporte de las asignaturas matriculadas.

RF_15. Gestionar ofertas.

15.1 Actualizar las ofertas de las matrículas en las asignaturas optativas.

15.2 Eliminar las ofertas de las matrículas en las asignaturas optativas.

15.3 Mostrar las ofertas de las matrículas en las asignaturas optativas.

15.4 Insertar las ofertas de las matrículas en las asignaturas optativas.

RF_16. Gestionar Usuarios.

16.1 Eliminar usuario.

16.2 Mostrar usuario.

RF_17. Gestionar Solicitudes.

17.1 Actualizar las solicitudes de las matrículas en las asignaturas optativas.

17.2 Mostrar las solicitudes de las matrículas en las asignaturas optativas.

17.3 Aceptar las solicitudes de las matrículas en las asignaturas optativas.

17.4 Cancelar las solicitudes de las matrículas en las asignaturas optativas.

2.4.3. Requisitos no funcionales de la aplicación web

- **Apariencia o interfaz externa**

Diseño orientado a crear una visión agradable para el usuario y con una navegación sencilla. Diseño intuitivo que permita a los usuarios usar cómoda y fácilmente la aplicación.

- **Requisitos de hardware**

Teniendo como base los requisitos mínimos de hardware de las herramientas y tecnologías seleccionadas para obtener un buen rendimiento se plantea:

Para uso del cliente: Procesador Intel Celeron a 1.8 GHZ (Gigahercio) o superior, 256 MB (Megabyte) de RAM ((*Random Access Memory o Memoria de Acceso Aleatorio*) o superior y 650 MB de espacio libre en disco.

Para uso del servidor: Procesador Intel Dual CPU 1.86 GHz o superior, 4 GB de RAM o superior y 40 GB de espacio libre en disco.

- **Requisitos de seguridad**

La seguridad del sistema desarrollado está basada en niveles de acceso sobre las funcionalidades y la información manejada por el sistema. Los principios que determinan la seguridad en el sistema son los siguientes.

- ✓ La seguridad se establecerá por roles que serán asignados a los usuarios que interactúen con el sistema, garantizando de esta forma que la información almacenada solo sea modificada y/o visualizada por los usuarios autorizados.

- **Requisitos de Usabilidad**

- ✓ El sistema está concebido para un número limitado de usuarios, sólo podrá ser utilizado por aquellas personas que estén autorizadas.
- ✓ El idioma definido para mostrar los mensajes y textos de la interfaz es el español.
- ✓ La navegabilidad no debe ser muy compleja, todas las funcionalidades deben ser rápidamente accesibles por el usuario empleándose la regla de los 3 *clicks* (3 *clicks rules*) la cual plantea que no se debería llegar a ninguna información clave de un sitio web en más de tres *clicks* (39).
- ✓ El sistema debe ser escalable para que al agregar nuevas funcionalidades no sean afectadas las que ya están funcionando.

- **Requisitos de Disponibilidad.**

El sistema debe estar disponible las 24 horas del día en el período que se realizarán las solicitudes, este período será definido por el administrador.

- **Portabilidad.**

La aplicación es multiplataforma (Windows/Linux).

2.4.4. Historias de Usuario

Una Historia de Usuario (HU) es una manera simple de describir una tarea concisa que aporta valor al usuario o al negocio. La diferencia más importante entre estas HU y los tradicionales documentos de especificación funcional se encuentra en el nivel de detalle requerido (40).

A continuación se muestra como quedan conformadas las HU Gestionar Facultades, Gestionar Período Lectivo, Gestionar Turnos Docentes y Gestionar Composición. El resto de las HU se encuentran en los anexos del presente documento (Ver Anexo 6).

Tabla 2 HU Gestionar Facultades

Historia de usuario	
ID: 1	**Nombre:** Gestionar Facultades
Usuario: Administrador	
Prioridad en el negocio: Alta	**Riesgo en desarrollo:** Medio
Estimación aproximada: 2 semanas	**Iteración asignada:** 1
Descripción: Se muestra un listado de todas las facultades de la UCI, las cuales se pueden activar o desactivar. Un estudiante puede acceder al sistema solo si la facultad a la que pertenece se encuentra activa.	
Observaciones: Responde a la HU Gestionar Facultades.	

Tabla 3 HU Gestionar Período Lectivo

Historia de usuario	
ID: 2	**Nombre:** Gestionar Período Lectivo
Usuario: Administrador	
Prioridad en el negocio: Alta	**Riesgo en desarrollo:** Medio
Estimación aproximada: 2 semanas	**Iteración asignada:** 1
Descripción: Permite actualizar y mostrar el período lectivo, permitiendo definir de esta manera el semestre actual en el que se encuentra el sistema.	
Observaciones: Responde a la HU Gestionar Período Lectivo.	

Tabla 4 HU Gestionar Turnos Docentes

Historia de usuario	
ID: 3	**Nombre:** Gestionar Turnos Docentes
Usuario: Administrador	
Prioridad en el negocio: Alta	**Riesgo en desarrollo:** Medio
Estimación aproximada: 2 semanas	**Iteración asignada:** 1
Descripción: Permite insertar, mostrar, actualizar y eliminar la información relacionada con los turnos docentes. De esta forma se definen los turnos docentes por los cuales se rige una jornada de clases en la institución.	
Observaciones: Responde a la HU Gestionar Turnos Docentes.	

Tabla 5 HU Gestionar Composición

Historia de usuario	
ID: 4	**Nombre:** Gestionar Composición
Usuario: Administrador	
Prioridad en el negocio: Alta	**Riesgo en desarrollo:** Medio
Estimación aproximada: 2 semanas	**Iteración asignada:** 1
Descripción: Permite insertar, mostrar, actualizar y eliminar información relacionada con la composición. De esta manera se puede definir la cantidad de asignaturas optativas a cursar por semestre.	
Observaciones: Responde a la HU Gestionar Composición.	

2.5. Fase de planificación

La planificación en XP responde dos preguntas claves del desarrollo de *software*: predecir qué se habrá terminado para la fecha de entrega, y determinar qué hacer después. Se hace énfasis en guiar al proyecto que es bastante directo en vez de predecir exactamente lo que se necesitará y cuánto tiempo tomará hacerlo que es bastante difícil (41).

2.5.1. Prioridad de las historias de usuarios

La asignación de prioridad a las historia de usuario por el cliente es lo que decide el orden en el cual estas se implementarán, siempre y cuando estén de acuerdo usuario y desarrollador (42). Las historias de usuario se clasifican en baja, media y alta, teniendo en cuenta las solicitudes del cliente y la lógica del negocio (43) quedando conformadas de la siguiente manera:

Tabla 6 Prioridad de las Historias de Usuario

Historia de usuario	Prioridad
Gestionar Facultades	Alta
Gestionar Período Lectivo	Alta
Gestionar Turnos Docentes	Alta
Gestionar Composición	Alta
Gestionar Años Académicos	Alta
Gestionar Disciplinas	Alta
Gestionar Asignaturas	Alta
Gestionar Ofertas	Alta
Gestionar Solicitudes	Alta
Realizar Solicitud	Alta
Gestionar Solicitudes Realizadas	Alta
Cargar Datos	Media
Gestionar Usuarios	Media
Generar Reportes	Media
Autenticar usuario	Baja
Mostrar información de usuario	Baja
Mostrar Asignaturas Cursadas	Baja

2.5.2. Estimación de esfuerzo

Las estimaciones de esfuerzo asociado a la implementación de las HU la establecen los programadores utilizando como medida el punto. Un punto, equivale a una semana ideal de programación. Las historias generalmente valen de 1 a 3 puntos. Por otra parte, el equipo de desarrollo mantiene un registro de la "velocidad" de desarrollo, establecida en puntos por iteración, basándose principalmente en la suma de puntos correspondientes a las historias de usuario que fueron terminadas en la última iteración (36). Especificando un tiempo estimado para la elaboración de cada una de las HU en base a una semana de 5 días y un día de 8 horas (Ver Anexo7).

2.5.3. Plan de Entrega

Es una práctica en donde el cliente presenta las características deseadas a los programadores, y los programadores estiman la dificultad. Teniendo las estimaciones de costo, y sabiendo la importancia de las características, el cliente establece un plan para el proyecto. Los planes iniciales de entregas son necesariamente imprecisos: ni las prioridades ni las estimaciones son sólidas, y tampoco se conoce qué tan rápido trabaja el equipo hasta que empiece a trabajar. Sin embargo, incluso el primer plan de entrega es lo suficientemente preciso como para tomar decisiones (41) (Ver Anexo 8).

2.6. Fase de iteración

Esta fase incluye varias iteraciones sobre el sistema antes de ser entregado. En la primera iteración se puede intentar establecer una arquitectura del sistema que pueda ser utilizada durante el resto del proyecto. El cliente es el encargado de decidir que HU es seleccionada para cada iteración (36).

De los planteamientos anteriores se determina que para el desarrollo de la aplicación web en cuestión se necesita un total de tres iteraciones teniendo en cuenta la prioridad asignada a cada HU en correspondencia con las solicitudes del cliente. En la primera iteración se tomaron todas aquellas HU de alta prioridad de manera que el producto aún sin terminar pueda mostrar las funcionalidades del sistema, para la segunda iteración se tomaron las HU de media prioridad para ir incorporando las principales funcionalidades al sistema dejando para la tercera iteración las restantes HU de baja prioridad, finalizándose de esta manera todas las funcionalidades descritas para la aplicación y obteniéndose el producto final.

Tabla 7 Primera iteración

Iteración		
Numero: 1	**H.U:** Gestionar Facultades, Gestionar Período Lectivo, Gestionar Turnos Docentes, Gestionar Composición, Gestionar Años Académicos, Gestionar Disciplinas, Gestionar Asignaturas, Gestionar Ofertas, Gestionar Solicitudes, Realizar Solicitud, Gestionar Solicitudes Realizadas.	**Duración total : 22 semanas**
Descripción: Se establece la arquitectura a utilizar en el desarrollo del sistema. Se desarrollan las historias de usuario de prioridad alta, conformando un producto que aún sin terminar pueda mostrar las funcionalidades del sistema, dará una primera vista al cliente que suministrará su criterio para la incorporación de nuevos elementos y modificación de los existentes.		

Tabla 8 Segunda iteración

Iteración		
Numero: 2	**H.U:** Cargar Datos, Gestionar Usuarios, Generar Reportes	**Duración total: 3 semanas**
Descripción: Se desarrollan las historias de usuario de prioridad media. Se llevan a cabo los señalamientos hechos por el cliente sobre la primera iteración y al finalizar dicho proceso se contará con una aplicación con las funcionalidades descritas en estas historias de usuarios, incorporando las principales funcionalidades al sistema.		

Tabla 9 Tercera iteración

Iteración		
Numero: 3	**H.U:** Autenticar usuario, Mostrar información de usuario, Mostrar Asignaturas Cursadas	**Duración total: 3 semanas**
Descripción: Se desarrollan las historias de usuario de prioridad baja. Se llevan a cabo los señalamientos hechos por el cliente sobre la primera y segunda iteración. Al finalizar dicho proceso se contará con las funcionalidades descritas en estas historias de usuarios, finalizándose de esta manera todas las funcionalidades descritas para la aplicación y obteniéndose el producto final.		

2.7. Cronograma de liberación

El Cronograma de liberación está basado en las iteraciones definidas por el cliente y el equipo de desarrollo, las mismas cuentan con las HU que se asignaron en cada iteración mostrando como quedará conformado el sistema según se realicen estas iteraciones y las fechas de liberación de la aplicación web.

En la siguiente tabla se muestra cómo quedará conformado el Cronograma de liberación.

Tabla 10 Cronograma de liberación

Iteración	Fecha de liberación
Primera iteración	Noviembre del 2014 – Marzo del 2015
Segunda iteración	Marzo del 2015 – Abril del 2105
Tercera iteración	Abril del 2015 – Mayo del 2015

2.7.1. Tarjetas CRC (Class-Responsibility-Collaboration o Clases-Responsabilidad-Colaboración).

La utilización de tarjetas CRC es una técnica de diseño orientado a objetos propuesta por Kent Beck (introductor de la metodología de programación extrema).El objetivo de la misma es hacer, mediante tarjetas, un inventario de las clases que se necesitan para implementar el sistema y la forma en que van a interactuar, de esta forma se pretende facilitar el análisis y discusión de las mismas por parte de varios actores del equipo de proyecto con el objetivo de que el diseño sea lo más simple posible verificando las especificaciones del sistema (44).

A continuación se muestran un ejemplo de cómo quedaron conformadas las tarjetas CRC, el resto de las tarjetas CRC se encuentran en los anexos del presente documento (Ver Anexo 9).

Tabla 11 CRC Gestionar Facultades

Tarjeta CRC	
Clase: FacultadController.php	
Responsabilidades	**Colaboraciones**
Definir qué Facultad se encuentra activa en el sistema de esta manera se permitir el acceso de los usuarios al sistema.	

2.8. Diagrama de despliegue

Es un diagrama estructurado que muestra la arquitectura del sistema desde el punto de vista del despliegue (distribución) de los artefactos del *software* mostrando las relaciones físicas entre los componentes de *hardware* y *software* del sistema.

Los artefactos representan elementos concretos en el mundo físico que son el resultado de un proceso de desarrollo.

El destino de despliegue está generalmente representado por un nodo que es, o bien de los dispositivos de hardware o bien de algún entorno de ejecución de *software*. Los nodos pueden ser conectados a través de vías de comunicación para crear sistemas en red de complejidad arbitraria (45). En la siguiente figura puede visualizarse el diagrama de despliegue definido para la solución propuesta:

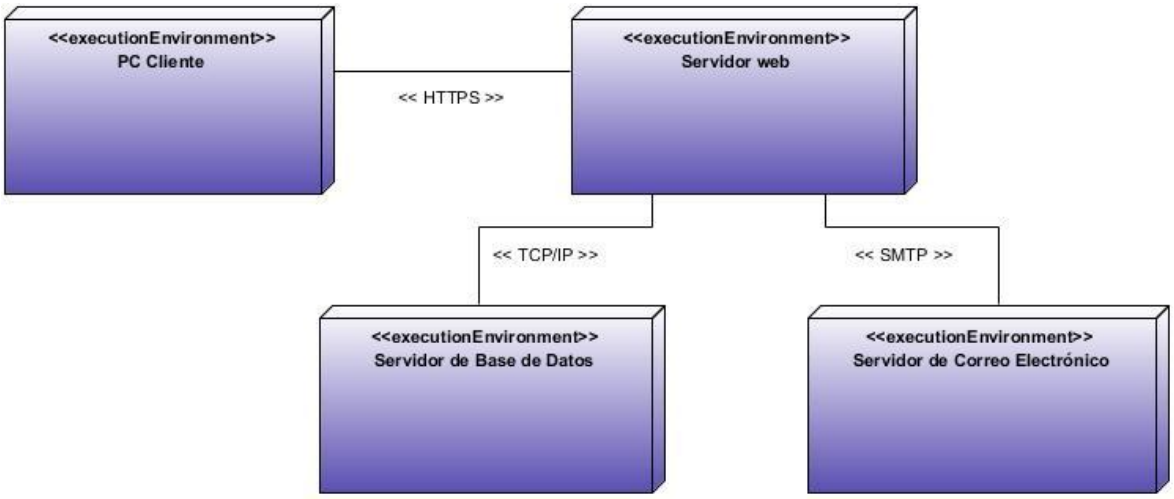


Figura 2 Diagrama de Despliegue

PC-Cliente: Ordenador desde donde los usuarios acceden a la aplicación. Tiene como función principal acceder al sistema e interactuar con el mismo según sus necesidades.

Servidor web: Ordenador donde se encuentra el Servidor web Apache, este será el lugar en que se gestione todo el contenido de la aplicación y las máquinas clientes acceden a través de un navegador web.

Servidor BD: El servidor de base de datos es el encargado de almacenar toda la información generada por el sistema, en él se encuentra ubicada la base de datos de donde los servicios que brinda el componente se sustenta de información.

HTTPS: Protocolo de transferencia de hipertexto seguro, por sus siglas en inglés, *Hypertext Transfer Secure Protocol* (HTTPS), es un protocolo de red basado en HTTP por lo que está orientado a transacciones sin estado, es decir, no guarda ninguna información sobre conexiones anteriores y sigue el esquema petición-respuesta entre un cliente y un servidor (46).

TCP/IP: Base de Internet, y sirve para enlazar computadoras que utilizan diferentes sistemas operativos. Protocolo utilizado para la conexión entre el servidor web y el servidor donde se encuentra ubicada la base de datos.

SMTP: (*Simple Mail Transfer Protocol* o Protocolo para la transferencia simple de correo electrónico) es un protocolo de comunicaciones basado en texto, y utilizado para intercambiar mensajes de correo electrónico, se emplea para gestionar el correo saliente (47).

2.9. Conclusiones

En este capítulo se realizó el diseño y el modelado de la propuesta del Sistema de Gestión de Asignaturas Optativas para la facultad 1, que perfecciona el proceso de gestión y almacenamiento de la información de la matrícula de las asignaturas optativas, lo que permitió concluir lo siguiente:

- El modelo del dominio permitió capturar y expresar la comprensión del proceso de matrícula de las asignaturas optativas.
- Se logró una descripción general de los requerimientos funcionales y no funcionales del sistema, así como de los actores que intervienen, los cuales fueron explicados a través de las tablas de las HU y las tarjetas CRC.
- El modelo de despliegue ilustra la comunicación de los distintos componentes de *hardware* en los cuales se divide la propuesta.

CAPÍTULO 3. IMPLEMENTACIÓN Y PRUEBAS DEL SISTEMA

3.0. Introducción

En este capítulo se plantean todas las líneas de descripción del diseño y construcción de la solución propuesta, basadas en el futuro funcionamiento del sistema a través de las tareas de ingeniería y las pruebas. Además, se fundamentan los principios de diseño que sustentan el entorno gráfico del Sistema de Gestión de Asignaturas Optativas.

3.1. Definición de la Arquitectura.

La arquitectura se refiere a un grupo de abstracciones y patrones que brindan un esquema de referencia útil para guiarnos en el desarrollo de *software* dentro de un sistema informático (48).

Se definió como arquitectura a utilizar la Cliente-Servidor

Esta arquitectura se divide en dos partes claramente diferenciadas, la primera es la parte del servidor y la segunda la de un conjunto de clientes. Normalmente el servidor es una máquina bastante potente que actúa de depósito de datos y funciona como un sistema gestor de base de datos (SGBD). Por otro lado los clientes suelen ser estaciones de trabajo que solicitan varios servicios al servidor. Ambas partes deben estar conectadas entre sí mediante una red (49). Una representación gráfica de este tipo de arquitectura sería la siguiente.

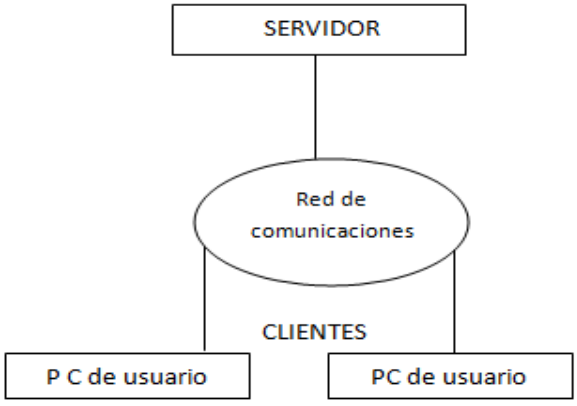


Figura 3 Arquitectura Cliente-Servidor (49).

3.1. Patrones de Diseño

Los patrones de diseño son un conjunto de prácticas de óptimo diseño que se utilizan para abordar problemas recurrentes en la programación orientada a objetos (50).

3.1.1. Patrones GOF (Gang Of Four o Pandilla de los Cuatro).

Son un conjunto de 23 patrones de diseño muy útiles durante el diseño de objetos. Estos se clasifican según el propósito para el que han sido definidos como: Patrones creacionales, estructurales y de comportamiento (51).

Dentro del grupo de Patrones estructurales se encuentra los Patrones Decorador y Fachada, dos de los cuales fueron utilizados en el desarrollo de la aplicación por lo que se describen a continuación:

Patrón Decorador

El patrón Decorador responde a la necesidad de añadir dinámicamente funcionalidad a un objeto. Esto permite no tener que crear sucesivas clases que hereden de la primera incorporando la nueva funcionalidad, sino otras que la implementan y se asocian a la primera (52).

Utilización en Symfony: El patrón de diseño decorador resuelve el problema al revés: la plantilla es decorada después de que el contenido es mostrado por una plantilla global, llamada *layout*.(5).

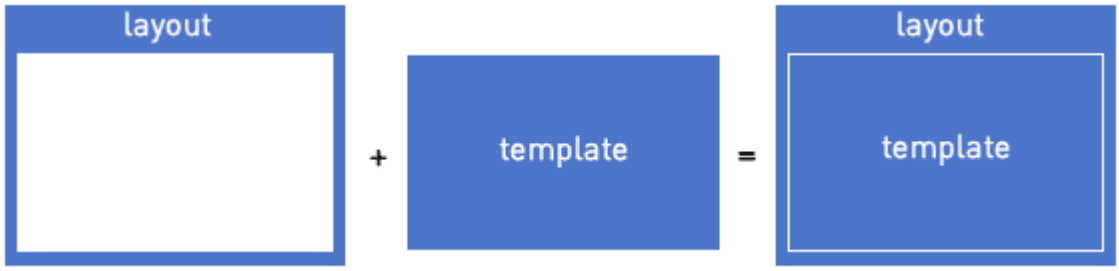


Figura 4 Patrón Decorador (*layout* en Symfony) (5)

Patrón Fachada

Una fachada es un objeto que ofrezca una sencilla interfaz que ocultará uno o varios sistemas más complejos y sus interacciones. Con este patrón se ofrece un acceso sencillo y se desacopla al máximo el sistema cliente (el que accede a la fachada) de los sistemas ocultos. Típicamente se utiliza en librerías o en sistemas diseñados en capas (53).

Utilización en Symfony: Conoce qué clases del subsistema son responsables de una determinada petición, y delega esas peticiones de los clientes a los objetos apropiados del subsistema.

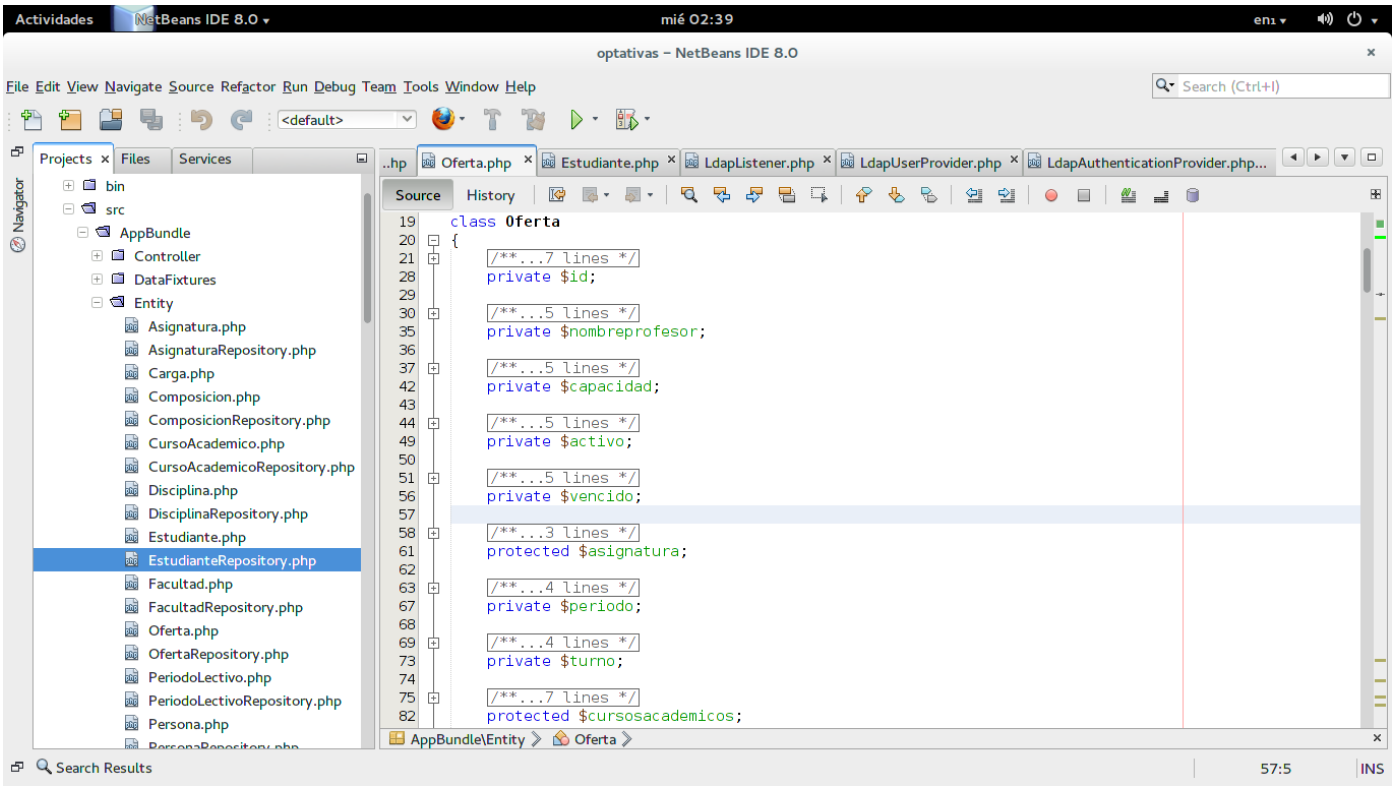


Figura 5 Patrón Fachada

3.1.2. Patrones GRASP (General Responsibility Assignment Software Patterns o Patrones Generales de Software para Asignar Responsabilidades).

Describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones (54).

Experto en Información.

Experto en información nos dice que la responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la información necesaria para crearlo. De este modo obtendremos un diseño con mayor cohesión y así la información se mantiene encapsulada, es decir, disminuye el acoplamiento (55).

Utilización en Symfony: Se evidencia en la clase Estudiante la cual hereda de la clase Persona agregándole atributos y funcionalidades que son propios de los Estudiantes, especializando el comportamiento de la clase Persona.

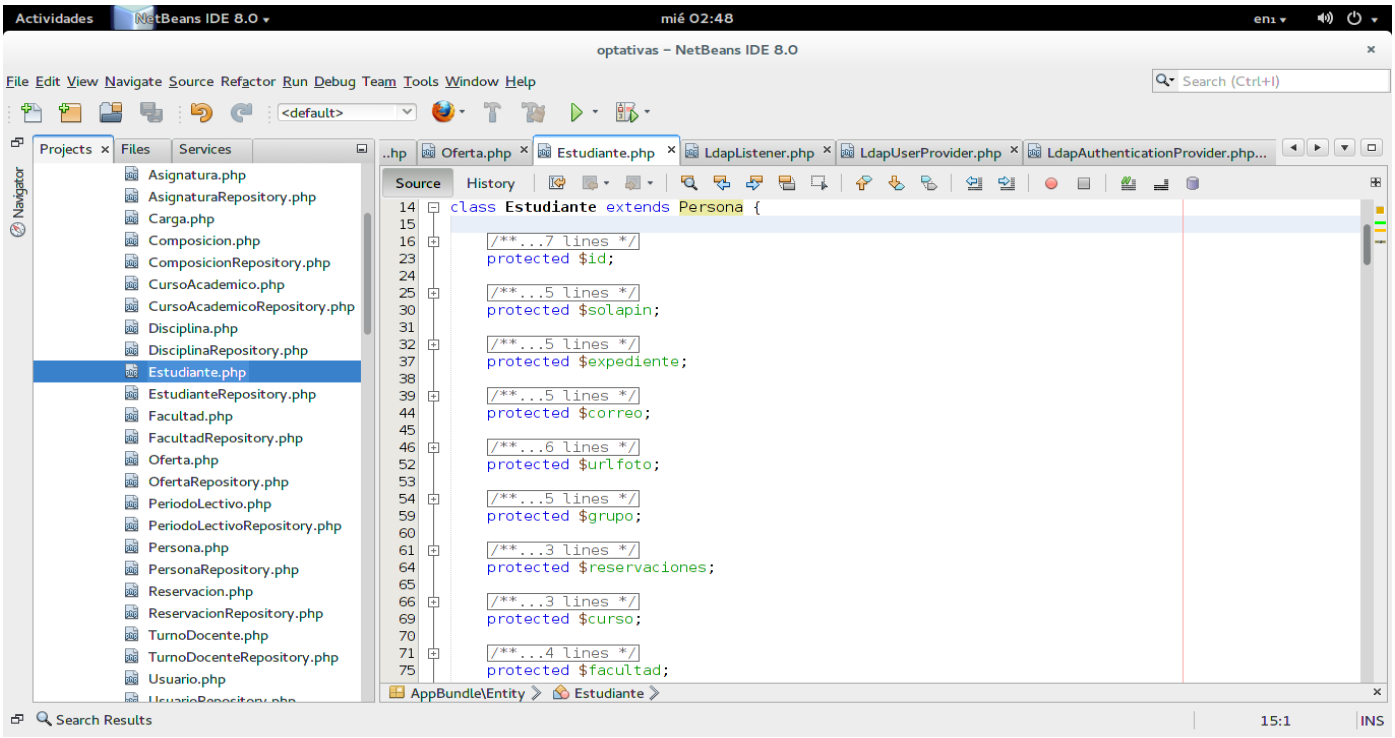


Figura 6 Patrón Experto

Creador

La creación de instancias es una de las actividades más comunes en un sistema orientado a objetos. En consecuencia es útil contar con un principio general para la asignación de las responsabilidades de creación. Si se asignan bien el diseño puede soportar un bajo acoplamiento, mayor claridad, encapsulamiento y reutilización. El patrón creador nos ayuda a identificar quién debe ser el responsable de la creación o instanciación de nuevos objetos o clases (56).

Utilización en Symfony: En las clases Controladoras se encuentran las acciones definidas para el sistema y se ejecutan en cada una de ellas. En dichas acciones se crean los objetos de las clases que representan las entidades, lo que evidencia que las clases son “creadoras” de dichas entidades.

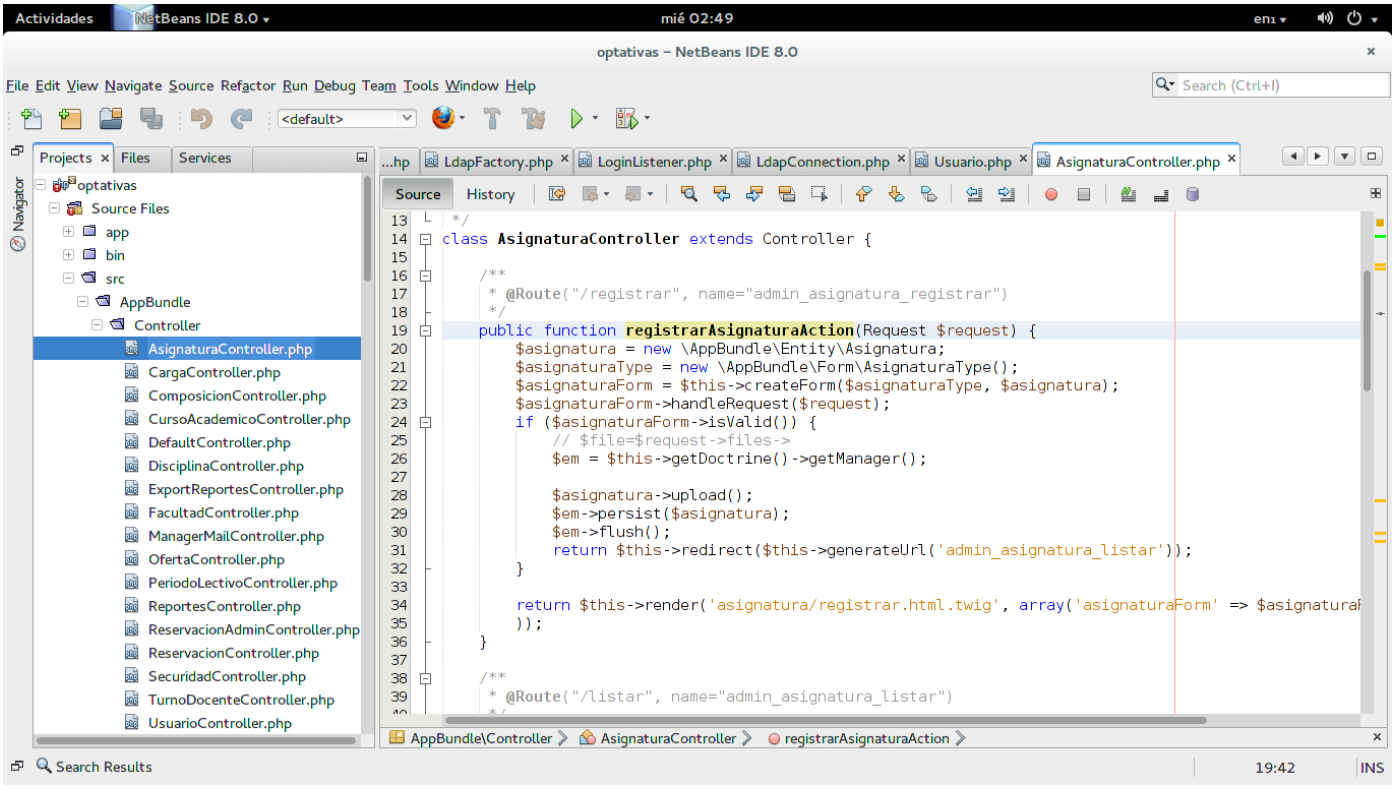


Figura 7 Patrón Creador

Controlador

El patrón controlador es un patrón que sirve como intermediario entre una determinada interfaz y el algoritmo que la implementa, de tal forma que es el controlador quien recibe los datos del usuario y quien los envía a las distintas clases según el método llamado (57).

Utilización en Symfony: La arquitectura del *framework* (MVC) ayuda desde el principio, pues existe una capa específicamente para los controladores, que son el núcleo del mismo. *Symfony* aplica el patrón "*Front Controller*" (Controlador frontal) y por tanto tiene una estructura bien organizada de controladores.

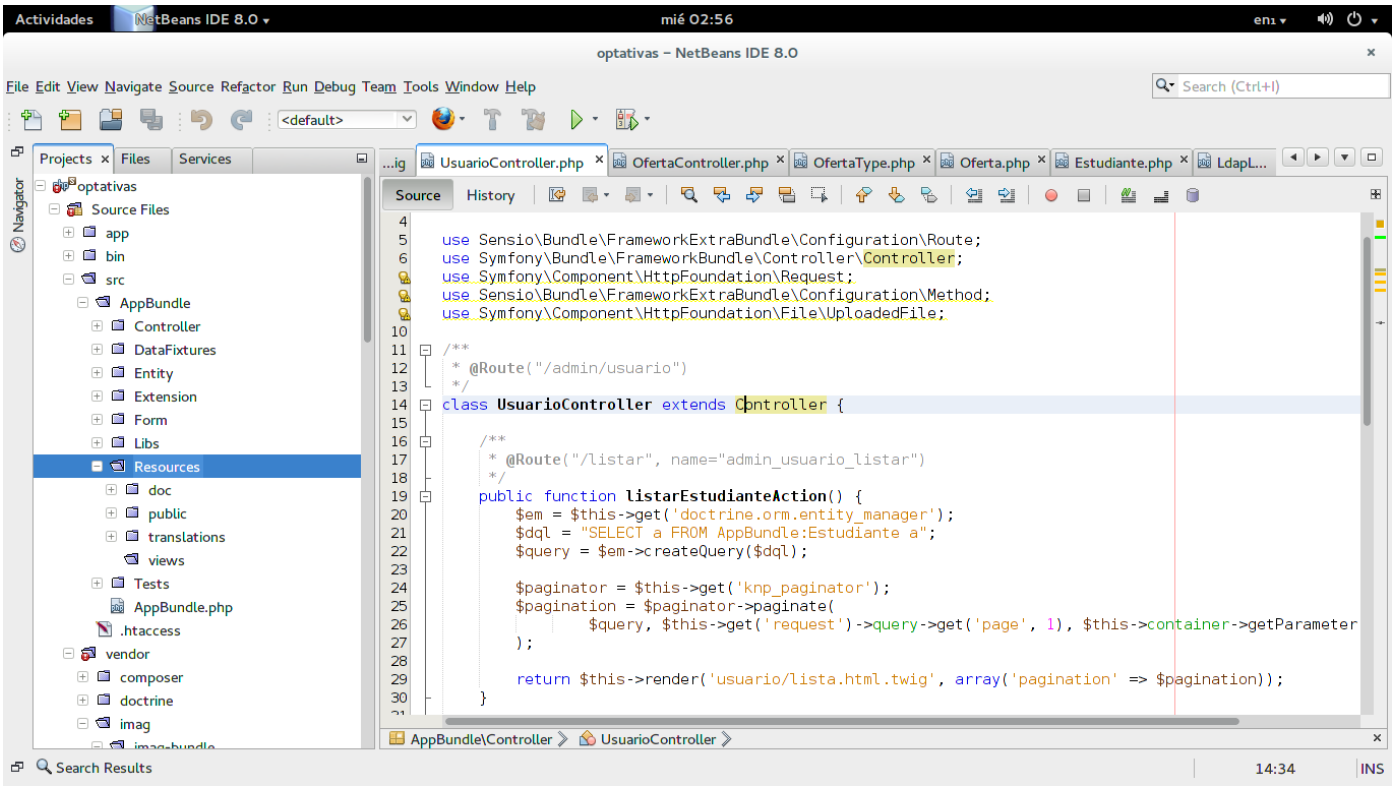


Figura 8 Patrón Controlador

Alta cohesión

El grado de cohesión mide la coherencia de la clase, esto es, lo coherente que es la información que almacena una clase con las responsabilidades y relaciones que ésta tiene. La alta cohesión indica que la información que almacena una clase debe de ser coherente y debe estar, en la medida de lo posible, relacionada con la clase (58).

Utilización en Symfony: Permite asignar responsabilidades con una alta cohesión, por ejemplo la clase *Actions* tiene la responsabilidad de definir las acciones para las plantillas y colabora con otras para realizar diferentes operaciones e instanciar objetos, es decir, está formada por diferentes funcionalidades que se encuentran estrechamente relacionadas proporcionando que el software sea flexible frente a grandes cambios.

Bajo acoplamiento

El grado de acoplamiento indica lo vinculadas que están unas clases con otras, es decir, lo que afecta un cambio en una clase a las demás y por tanto lo dependientes que son unas clases de otras.(58)

Utilización en Symfony: La clase *Action* hereda solamente de *sfActions* para lograr un bajo acoplamiento de clases.

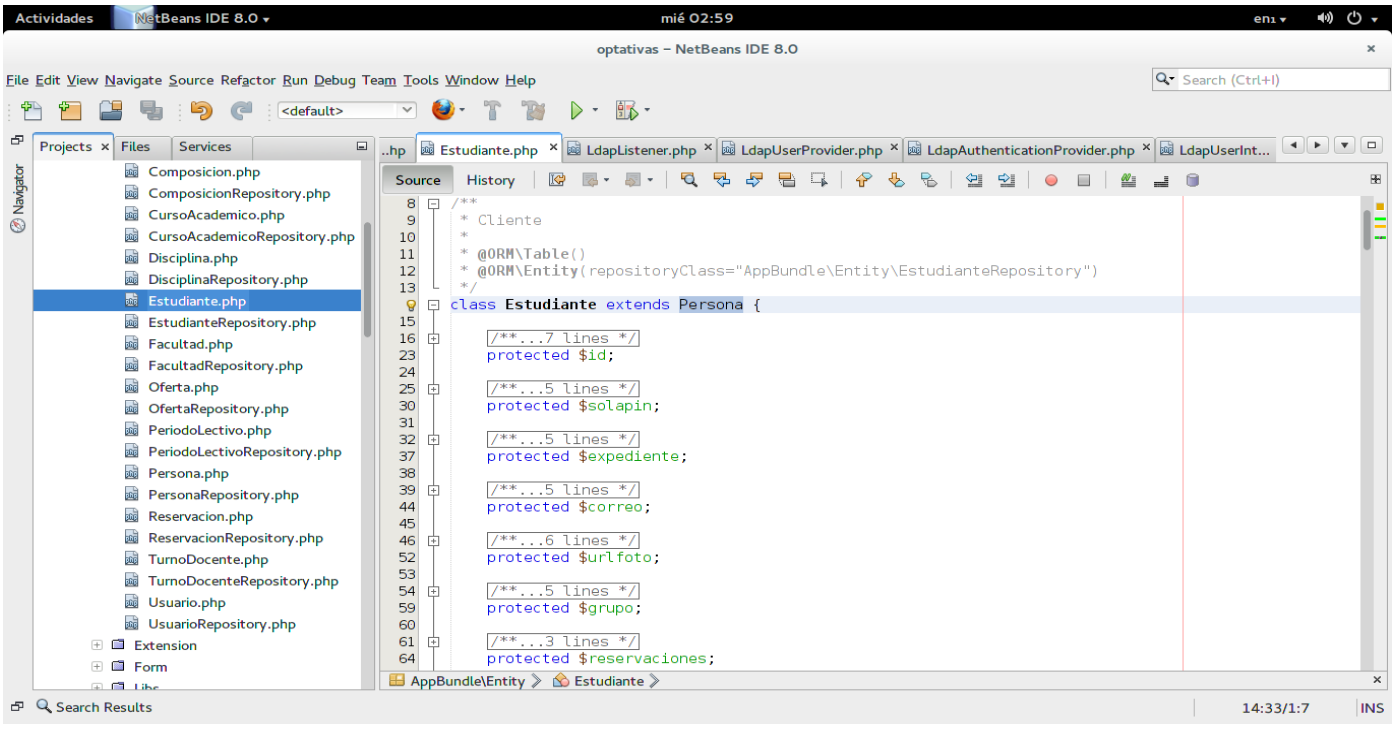


Figura 9 Patrón Bajo Acoplamiento

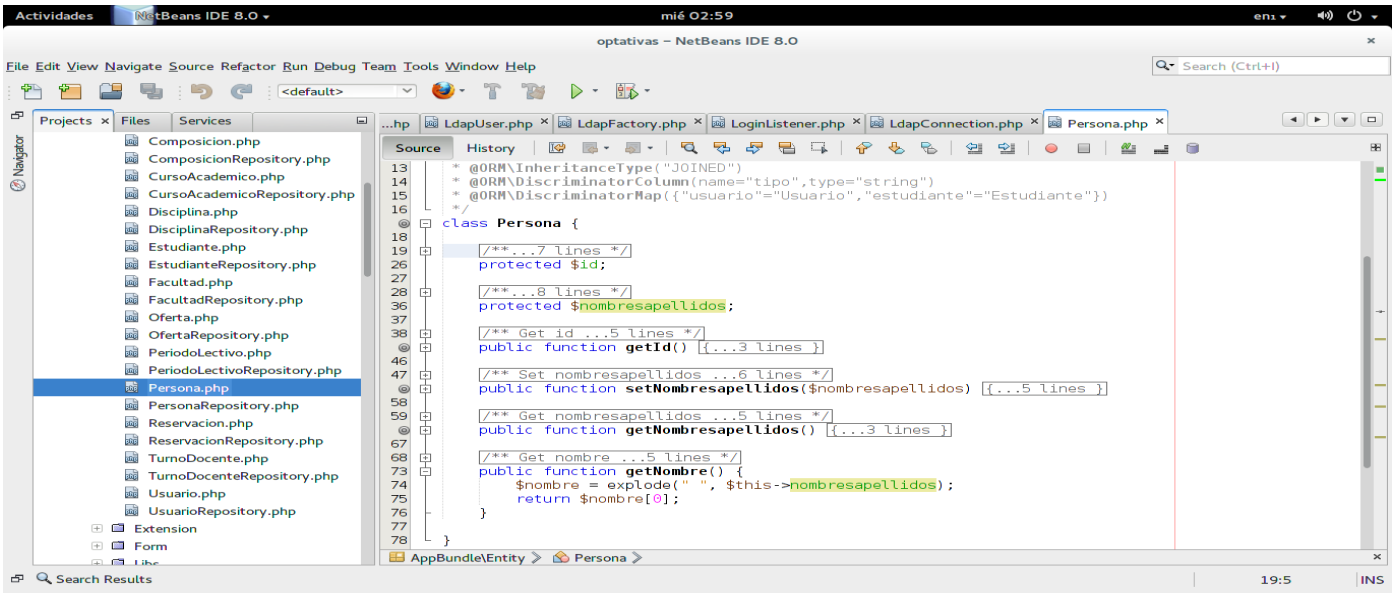


Figura 10 Patrón Alta Cohesión

3.1.3. Patrones y Estilos Arquitectónicos

MVC (*Model-View-Controller* o Modelo-Vista-Controlador)

Es un patrón de diseño que separa los datos, la lógica de negocios y las interfaces de usuario. Como su nombre lo dice, está separado en tres componentes: modelo, controlador y vista. Está basado en la ideología de separación de conceptos y cumple perfectamente con los objetivos de los patrones de diseño (59).

Symfony está basado en un patrón clásico del diseño web conocido como arquitectura MVC, que está formado por tres niveles

1. El Modelo representa la información con la que trabaja la aplicación.
2. La Vista transforma el modelo en una página web que permite al usuario interactuar con ella.
3. El Controlador se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista.

La arquitectura MVC separa la lógica de negocio y la presentación por lo que se consigue un mantenimiento más sencillo de las aplicaciones. Si por ejemplo una misma aplicación debe ejecutarse tanto en un navegador estándar como un navegador de un dispositivo móvil, solamente es necesario crear una vista nueva para cada dispositivo; manteniendo el controlador y el modelo original. El controlador se encarga de aislar al modelo y a la vista de los detalles del protocolo utilizado para las peticiones (HTTP, consola de comandos y email). El modelo se encarga de la abstracción de la lógica relacionada con los datos, haciendo que la vista y las acciones sean independientes de, por ejemplo, el tipo de gestor de bases de datos utilizado por la aplicación. La siguiente figura muestra el funcionamiento del patrón MVC.

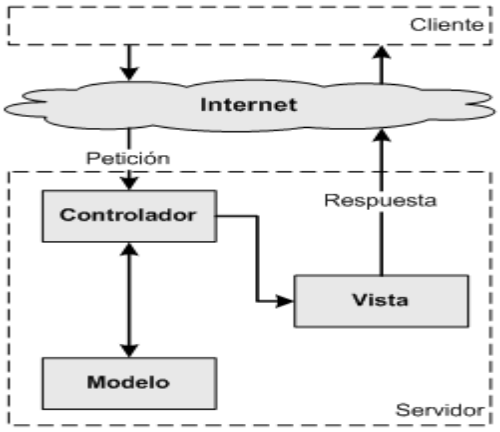


Figura 11 Funcionamiento del patrón MVC (60).

Symfony toma lo mejor de la arquitectura MVC y la implementa de forma que el desarrollo de aplicaciones sea rápido y sencillo. Utilizando un total de siete componentes (60).

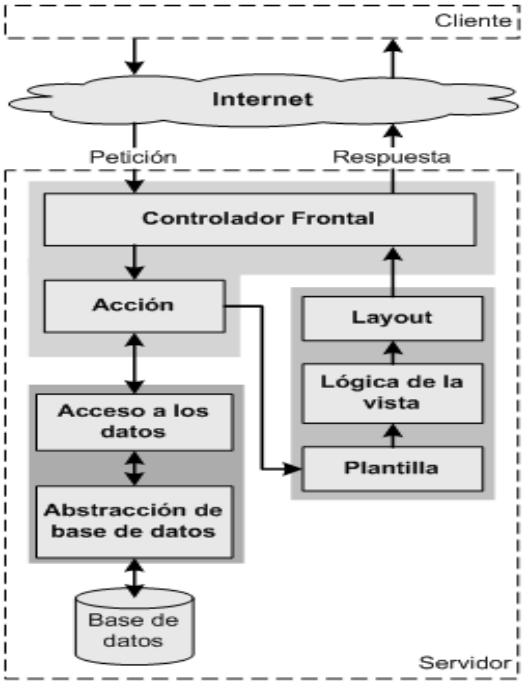


Figura 12 Flujo de trabajo de Symfony (60).

3.2. Tareas de Ingeniería (TI)

Una tarea de ingeniería se define como un conjunto de actividades en los cuales, utilizando técnicas y herramientas, se analiza un problema y se concluye con la especificación de una solución. La ingeniería de requisitos es el proceso de desarrollar una especificación de *software* (61).

A continuación una de las tareas definidas para el desarrollo de la presente investigación, el resto de las tareas se encuentran en los anexos del presente documento (Ver Anexo10).

Tabla 12 TI_ Diseñar la vista de Gestionar Facultades

Tareas de Ingeniería		
ID: 1	**H.U:** Gestionar Facultades	**Iteración :** 1
Nombre: Diseñar la vista de Gestionar Facultades		
Tipo: Desarrollo	**Puntos estimados:** 2	
Fecha inicio: Noviembre del 2014	**Fecha fin:** Marzo del 2015	
Responsable: Roberto Noel Acosta Cabrera.		
Descripción: Desarrollar la vista Gestionar Facultades, centrándose en la información generada por el sistema pertenecientes a las facultades.		

Tabla 13 TI_ Conectar Gestionar Facultades con la Base de datos.

Tareas de Historia de Usuario		
ID: 1.1	**H.U:** Gestionar Facultades	**Iteración :** 2
Nombre: Conectar Gestionar Facultades con la Base de datos.		
Tipo: Desarrollo	**Puntos estimados:** 2	
Fecha inicio: Noviembre del 2014	**Fecha fin:** Marzo del 2015	
Responsable: Roberto Noel Acosta Cabrera.		
Descripción: Conectar la vista de Gestionar Facultades con la BD y validar los datos.		

3.3. Pruebas

Dentro de las buenas prácticas de la metodología XP, se encuentra la de llevar a cabo un Desarrollo Guiado por Pruebas (TDD por su siglas en inglés), pues de esta manera se reduce el número de errores no detectados, así como el tiempo entre la introducción de estos en el sistema y su descubrimiento. Es por esta razón que el sistema en desarrollo está siendo probado constantemente. La metodología XP divide las pruebas en dos grupos fundamentales, las pruebas unitarias y las pruebas de aceptación (62).

3.3.1. Pruebas de aceptación (PA)

Las pruebas de aceptación o *test* de aceptación permiten comprobar que el *software* cumple con un requisito de negocio. Una funcionalidad escrita con el lenguaje del cliente pero que puede ser ejecutada por la máquina. Estas pruebas son creadas a partir de las historias de usuario. Las mismas son el punto de partida del desarrollo en cada iteración (63). Las pruebas de aceptación permiten además, comprobar que la funcionalidad que se está probando sea la esperada por el cliente. Este tipo de pruebas funcionan como una caja negra, pues cada una de ellas representa una salida esperada del sistema, donde es responsabilidad del cliente verificar la corrección de las pruebas y tomar decisiones acerca de las mismas. A continuación, se muestra una de las plantillas que se elaboran para realizar estas pruebas. El resto se encuentran en los anexos del presente documento (Ver Anexo 11).

Tabla 14 PA_ Gestionar Facultades

Prueba de aceptación		
Código: H.U1_P.A1	**H.U:** Gestionar Facultades	
Nombre: Gestionar Facultades		
Descripción: Permite habilitar y deshabilitar las facultades para restringir el acceso a la aplicación.		
Condiciones de Ejecución: En caso de encontrarse deshabilitadas permita habilitarlas y viceversa.		
Entrada / Pasos de ejecución: Se muestra un listado con las facultades y el estado en el que se encuentra luego se puede habilitar o deshabilitar las mismas.		
Resultados esperados:	**Escenario positivo:** Datos correctos	**Escenario negativo:** Datos incorrectos
	Son habilitadas o deshabilitadas las facultades	Muestra un mensaje de error.
Evaluación: Satisfactoria		

3.3.2. Prueba Funcional (PF)

Objetivo:

Se asegura el trabajo apropiado de los requisitos funcionales, incluyendo la navegación, entrada de datos, procesamiento y obtención de resultados.

Las metas de estas pruebas son:

1. Verificar el procesamiento, recuperación e implementación adecuada de las reglas del negocio.
2. Verificar la apropiada aceptación de datos.

Enfoque:

Los requisitos funcionales y las reglas del negocio.

Técnica:

Caja Negra. Se ejecuta cada función, usando datos válidos e inválidos, para verificar lo siguiente:

1. Que se aplique apropiadamente cada regla de negocio.
2. Que los resultados esperados ocurran cuando se usen datos válidos.
3. Que sean desplegados los mensajes apropiados de error y precaución cuando se usan datos inválidos.(64)

En el Anexo 12 se muestran Las Pruebas de Acceso al Sistema. (Ver Anexo 12).

3.3.3. Prueba de Seguridad (PS)

Objetivo

Nivel de Seguridad de la Aplicación:

- Verifica que un actor solo pueda acceder a las funciones y datos que su usuario tiene permitido.

Nivel de Seguridad del Sistema:

- Verificar que solo los actores con acceso al sistema están habilitados para acceder a las funcionalidades.

Áreas

Seguridad del sistema, incluyendo acceso a datos o funciones de negocios. Seguridad del sistema, incluyendo ingresos y accesos remotos al sistema.

Garantiza

Que los usuarios están restringidos a funciones específicas o su acceso está limitado únicamente a los datos que están autorizados a acceder. Que solo aquellos usuarios autorizados a acceder al sistema son capaces de ejecutar las funciones del sistema.

Técnicas

Identificar cada tipo de usuario y las funciones y datos a los que se debe autorizar.

En el Anexo 13 se muestran Las Pruebas de Seguridad. (Ver Anexo 13).

3.3.4. Pruebas de carga y estrés

El rendimiento de la aplicación es un elemento clave en la aceptación de la misma, cualquiera que esta sea. Este es totalmente dependiente de varios factores, entre los que se incluyen, la calidad de la conexión a la red así como el hardware que se utilice.

El Apache JMeter fue la herramienta utilizada para realizar las pruebas de rendimiento a la aplicación web. Esta aplicación, entre su gama de funcionalidades recoge un número importante de datos sobre los resultados capturados.

Para la realización de las pruebas de carga y estrés se utilizó la herramienta Apache Jmeter. Las pruebas se realizaron desde una computadora con 2GB de RAM, microprocesador Intel Dual CPU 1.8.6 GHz y sistema operativo Ubuntu 14.04. A continuación, se describen las variables que miden el resultado de las pruebas de carga y estrés realizadas al sistema.

Muestra: Cantidad de peticiones realizadas para cada URL.

Media: Tiempo promedio en milisegundos en el que se obtienen los resultados.

Mediana: Tiempo en milisegundos en el que se obtuvo el resultado que ocupa la posición central.

Min: Tiempo mínimo que demora un hilo en acceder a una página.

Max: Tiempo máximo que demora un hilo en acceder a una página.

Línea 90 %: Máximo tiempo utilizado por el 90 % de la muestra, al resto de la misma le llevo más tiempo.

% Error: Por ciento de error de las páginas que no se llegaron a cargar de manera satisfactoria.

Rendimiento (Rend): El rendimiento se mide en cantidad de solicitudes por segundo.

Kb/Seg: El rendimiento se mide en cantidad de kilobytes por segundo.

Como se muestra en la siguiente tabla, se simularon las peticiones realizadas al sistema por un total de 500, 650 y 800 usuarios simultáneamente, obteniéndose los siguientes resultados:

Tabla 15 Prueba de carga y estrés

Usuarios	Muestra	Media	Mediana	Línea 90 %	Mín	Máx	% Error	Rend.	Kb/seg
500	3800	2269	2384	2652	3	2576	0	73.4	664.7
650	43000	3382	3673	3440	4	456781	0.13	68.3	673.6
850	54624	5130	5372	8432	5	553622	0.39	46.2	327.9

Las pruebas realizadas muestran que el sistema es capaz de responder a 3800 peticiones de 500 usuarios conectados simultáneamente en un tiempo promedio de 2269 milisegundos (2.3 segundos aproximadamente) con 0 % de error. Esto evidencia que el sistema puede procesar la carga esperada.

Por otra parte, se realizaron 43000 peticiones iniciadas por 650 usuarios y en este caso el sistema respondió en 3382 milisegundos (3.4 segundos aproximadamente) como tiempo promedio. No fue capaz de responder correctamente el 0.13 % de las peticiones realizadas.

Por último se realizaron 54624 peticiones iniciadas por 850 usuarios y en este caso el sistema respondió en 5130 milisegundos (5.1 segundos aproximadamente) como tiempo promedio. No fue capaz de responder correctamente el 0.38 % de las peticiones realizadas.

3.4. Resultado de las pruebas

Prueba Funcional

Uno de los detalles a no pasar por alto al momento de realizar las pruebas son las no conformidades, las cuales se traducen en los errores encontrados y funcionalidades no deseadas detectadas en cada iteración de pruebas. Al final de cada iteración se muestra una versión funcional del *software* de forma que se pueda detectar aquellas no conformidades que serán corregidas al inicio de la siguiente iteración.

La figura 8 representa un gráfico de barras con los resultados de las pruebas desarrolladas en las 3 iteraciones al sistema, asociadas a las no conformidades detectadas en cada una de ellas.

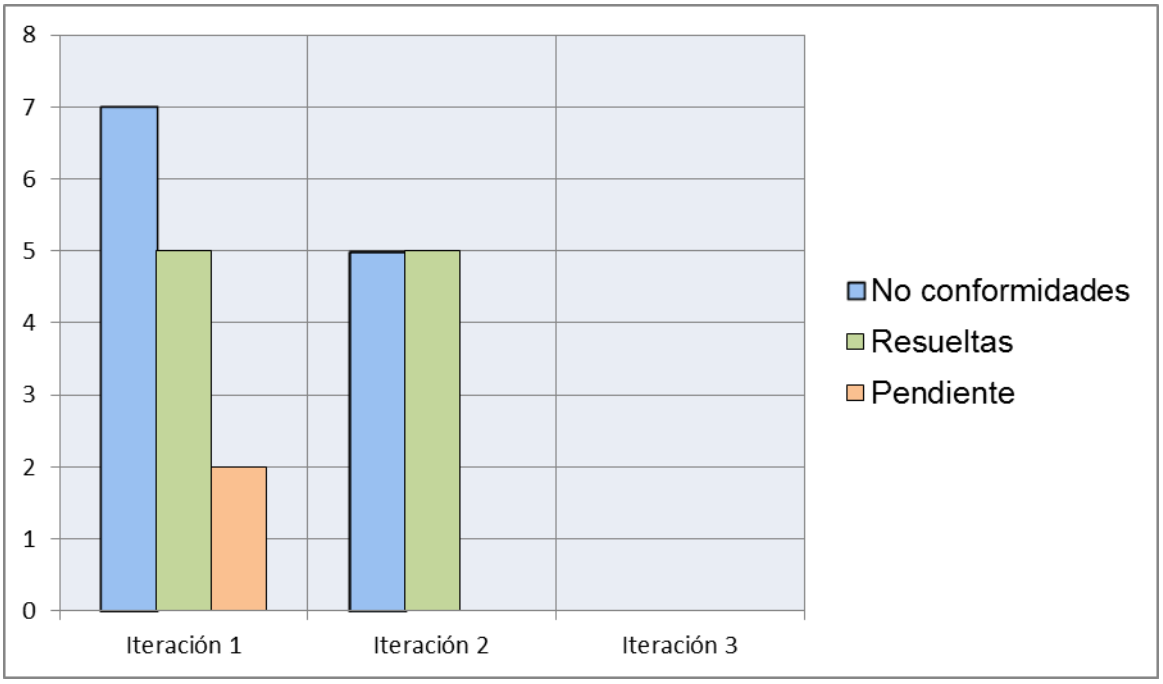


Figura 13 Resultado de las no conformidades

A continuación se listan las no conformidades encontradas en cada una de ellas:

Tabla 16 Registro de no conformidades

No conformidades	
Iteración:	**Descripción:**
Primera	1. Errores ortográficos en los mensajes de la pantalla de autenticación. 2. No se especifica en el mensaje mostrando al usuario los errores que tiene el formulario. 3. Permite insertar números en el campo de nombre de usuario. 4. Retardo en la carga de las pestañas. 5. Error al cargar los datos del archivo Excel. 6. Cambiar el mensaje 404. 7. Errores de diseño en algunas vistas.
Segunda	1. Pérdida de datos. 2. Errores de concordancia en los mensajes. 3. Mensajes de error pocos intuitivos. 4. Error al cargar los datos del archivo Excel. 5. Cambiar el mensaje 404.
Tercera	No se encuentran no conformidades en el sistema.

Como parte de la metodología XP, las no conformidades encontradas en cada iteración son las primeras tareas a resolver de la iteración siguiente, siendo el cliente el encargado de ordenarlas por prioridad. Algunas de ellas al no ser críticas, son arrastradas a la siguiente iteración. Llevando a cabo este proceso, se logran minimizar los niveles de aceptación de errores. De esta manera quedaron resueltas las no conformidades detectadas en la aplicación desarrollada.

Prueba de Seguridad

Los resultados fueron satisfactorios ya que las pruebas realizadas, permitieron comprobar que existe un correcto manejo de la seguridad en el sistema.

Prueba de Aceptación

Los resultados fueron satisfactorios ya que las pruebas realizadas, fueron avaladas por el cliente.

3.5. Conclusiones

En este capítulo se realizó la implementación y validación de la propuesta del Sistema de Gestión de Asignaturas Optativas para la facultad 1, lo que permitió concluir lo siguiente:

- La arquitectura a utilizar es la Cliente-Servidor y los patrones de diseño que se utilizan son los que está dentro del GOF (Patrón Decorador y Patrón Fachada), así como los patrones que están en el GRASP (Creador, Experto en Información, Controlador, Bajo acoplamiento, Alta cohesión).
- Las tareas de ingeniería llevadas a cabo sirvieron como soporte organizativo en el desarrollo del sistema.
- El desarrollo de las pruebas realizadas, permitieron validar el correcto funcionamiento del sistema, así como la identificación de las no conformidades.

CONCLUSIONES GENERALES

A partir del desarrollo de la presente investigación se arribó a las siguientes conclusiones:

1. Mediante la exploración de la realidad, el análisis documental, las entrevistas y las encuestas se comprobó la validez del problema que se platea en la investigación.
2. La fundamentación de los referentes teórico-metodológicos del proceso de gestión y almacenamiento de la información de la matrícula de las asignaturas optativas, permitió identificar las tendencias y tecnologías, así como las herramientas, lenguajes y metodología para el desarrollo de la solución.
3. El estudio y el análisis de los sistemas de homólogos, permitió obtener los conocimientos y la información necesaria para comprender el objeto que se investigó y sentar las bases de la investigación.
4. La realización de las pruebas al Sistema de Gestión de Asignaturas Optativas, determinó el correcto funcionamiento de cada una de las funcionalidades implementadas.
5. El desarrollo del Sistema de Gestión de Asignaturas Optativas perfecciona el proceso de gestión y almacenamiento de la información de la matrícula de las asignaturas optativas de la Facultad 1 de la Universidad de las Ciencias Informáticas.
6. El objetivo elaborado en el diseño de la investigación fue cumplido, comprobándose la hipótesis como respuesta al problema científico que originó la investigación, tributando directamente al proceso de gestión y almacenamiento de la información de la matrícula de las asignaturas optativas.

RECOMENDACIONES

1. Se mantenga constante actualización y mantenimiento del sistema de tal modo que pueda brindar cada una de las funcionalidades implementadas de la forma óptima.
2. Se realice un despliegue de la solución informática a cada una de las facultades de la universidad logrando informatizar el proceso de matrícula de las asignaturas optativas.
3. Poner en explotación el sistema permitiendo que sea accesible por el estudiantado desde cualquier lugar de la universidad.

REFERENCIAS BIBLIOGRÁFICAS

1. Definición de Asignatura. *Definición ABC* [online]. Available from: http://www.definicionabc.com/general/asignatura.php

2. Reglamento para el Trabajo Docente y Metodológico en la Educación Superior. *Resolución 210/07*. 2007. p. 217.

3. LAPUENTE, Chusa Lamarca and LAPUENTE, María Jesús Lamarca. Lenguaje UML. [online]. Available from: http://www.hipertexto.info/documentos/uml.htm

4. MIGUEL DÍAZCANEL BERMÚDEZ. *RESOLUCIÓN No. 120 /10* [online]. Available from: http://files.sld.cu/sccs/files/2010/09/reglamento-org-docente-mes-res-120-2010.pdf

5. SYMFONY DOCUMENTATION. Symfony. [online]. Available from: http://symfony.com/legacy/doc/jobeet/1_4/es/04?orm=Doctrine

6. Gestión. *Definición MX* [online]. [Accessed 24 October 2014]. Available from: http://definicion.mx/gestion/Definición de Gestión

7. DocCF, Software de Gestión Escolar. *Grupo CF Developer | DocCF, Software de Gestión Escolar* [online]. Available from: http://www.grupocfdeveloper.com/

8. Universidad Industrial de Santander. Máster Universitario en Gestión Ambiental de Sistemas Hídricos. [online]. Available from: http://www.unican.es/WebUC/catalogo/planes/detalle_od.asp?id=43

9. DocCF, Software de Gestión Escolar. [online]. Available from: http://www.grupocfdeveloper.com/

10. MSC. LILIA ESTER RODRÍGUEZ CHÁVEZ, Lic. Diana Margarita Rey Kaba. *SISTEMA AUTOMATIZADO DE GESTIÓN DE LA MAESTRÍA INFORMÁTICA EN SALUD* [online]. Available from: http://www.rcim.sld.cu/revista_23/articulo_pdf/sistema.pdf

11. Sistema de Gestión universitaria - Monografias.com. [online]. Available from: http://www.monografias.com/trabajos92/gestion-universitaria/gestion-universitaria.shtml

12. Desarrollo de Software. [online]. Available from: http://www.neosoft.es/servicios-informaticos/desarrollo-de-software/

13. Aplicaciones web Vs Aplicaciones de escritorio | Blog de informática y tecnología | Consultoría Informática. [online]. Available from: http://www.webprogramacion.com/356/blog-informatica-tecnologia/aplicaciones-web-vs-aplicaciones-de-escritorio.aspx

14. Desarrollo de Software. [online]. Available from: http://www.neosoft.es/servicios-informaticos/desarrollo-de-software/

15. ZAMORA, Marlen. Aplicaciones Web: VENTAJAS Y DESVENTAJAS DE APLICACIONES WEB. *Aplicaciones Web* [online]. 6 October 2012. Available from: http://unidad6aplicacionesweb1.blogspot.com/2012/10/blog-post.html

16. Servidor web y PHP. *MBlog* [online]. Available from: http://manuelbaronetti.com.ar/blog/2013/02/servidor-web-y-php/

17. CIBERAULA. Una Introducción a Apache. [online]. Available from: http://linux.ciberaula.com/articulo/linux_apache_intro

18. LEANDRO ALEGSA. Definicion de PHP. *Diccionario deinformática y Tecnología* [online]. 2015 1998. Available from: http://www.alegsa.com.ar/Dic/php.php

19. Qué es PHP. *DesarrolloWeb.com* [online]. Available from: http://www.desarrolloweb.com/articulos/392.php

20. HTML. *Mozilla Developer Network* [online]. Available from: https://developer.mozilla.org/es/docs/Web/HTML

21. Definición de html — Definicion.de. *Definición.de* [online]. Available from: http://definicion.de/html/

22. MASADELANTE.COM. Definición de CSS. [online]. Available from: https://www.masadelante.com/faqs/css

23. ¿Qué es Javascript? *Maestros del Web* [online]. Available from: http://www.maestrosdelweb.com/que-es-javascript/

24. DICE, LISSET ALVARENGA. ¿Qué es PostgreSQL? *Microbuffer* [online]. Available from: http://microbuffer.wordpress.com/2011/05/04/que-es-postgresql/

25. PgAdmin III. [online]. Available from: http://www.guia-ubuntu.com/index.php/PgAdmin_III

26. METODOLOGIAS PARA DESARROLLO DE SOFTWARE. [online]. Available from: http://procesosdesoftware.wikispaces.com/METODOLOGIAS+PARA+DESARROLLO+DE+SOFTWARE

27. XP - Extreme Programing Ingenieria de Software. [online]. Available from: http://ingenieriadesoftware.mex.tl/52753_XP---Extreme-Programing.html

28. Ciclo de vida de un proyecto XP. [online]. Available from: http://oness.sourceforge.net/proyecto/html/ch05s02.html

29. *Introducción a Herramientas CASE y System Architect* [online]. Available from: http://users.dsic.upv.es/asignaturas/eui/mtp/doc-practicas/intro_case_SA.pdf

30. Visual Paradigm para UML. [online]. Available from: http://www.software.com.ar/visual-paradigm-para-uml.html

31. NetBeans IDE entorno de desarrollo para lenguajes como Java PHP C/C++ Groovy. [online]. Available from: http://www.genbetadev.com/herramientas/netbeans-1

32. JAVIER, Gutierrez. *¿Qué es un framework web?* [online]. Available from: http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf

33. 1.1. Symfony en pocas palabras (Symfony 1.4, la guía definitiva). [online]. Available from: http://librosweb.es/libro/symfony_1_4/capitulo_1/symfony_en_pocas_palabras.html

34. ¿Qué es Bootstrap y cómo funciona en el diseño web? *Chucherías* [online]. Available from: http://www.arweb.com/chucherias/editorial/%c2%bfque-es-bootstrap-y-como-funciona-en-el-diseno-web.htm

35. Modelo de Dominio. *Tecnología y Synergix* [online]. Available from: https://synergix.wordpress.com/2008/07/10/modelo-de-dominio/

36. LETELIER, Patricio and PENADÉS, Mª Carmen. Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP). *www.cyta.com.ar/ta0502/v5n2a1.htm* [online]. 15 April 2006. Available from: http://www.cyta.com.ar/ta0502/v5n2a1.htm

37. ING.DANAY PÉREZ RAMÍREZ. *METODOLOGÍAS ÁGILES.¿CÓMO DESARROLLO UTILIZANDO XP?* [online]. 2008. Available from: http://www.google.com.cu/url?sa=t&rct=j&q=&esrc=s&source=web&cd=14&ved=0CF8QFjAN&url=http%3A%2F%2Fccia.cujae.edu.cu%2Findex.php%2Fsiia%2Fsiia2008%2Fpaper%2Fdownload%2F1174%2F246&ei=NUjlVILREs2xyATcoYLwBA&usg=AFQjCNEz2tUXfZGa0x15XKW3rj54uP4bng&bvm=bv.85970519,d.aWw&cad=rja

38. 3. Técnicas para Identificar Requisitos Funcionales y No Funcionales - Metodología Gestión de Requerimientos. [online]. Available from: https://sites.google.com/site/metodologiareq/capitulo-ii/tecnicas-para-identificar-requisitos-funcionales-y-no-funcionales

39. Usandolo.com » Principios de usabilidad: La regla de los 3 clicks. [online]. Available from: http://usandolo.com/principios-de-usabilidad-la-regla-de-los-3-clicks/

40. Historias de usuario¿Por qué? ¿Qué son? ¿Cómo son? [online]. 12:12:35 UTC. Available from: http://es.slideshare.net/MiquelMora/historias-de-usuario

41. Una introducción a Extreme Programming. [online]. Available from: http://www.dosideas.com/noticias/metodologias/822-una-introduccion-a-extreme-programming.html

42. PENADÉS, Mª Carmen. Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP). . 2008. Departamento de Sistemas Informáticos y Computación (DSIC) Universidad Politécnica de Valencia (UPV)

43. A propósito de programación extrema XP (eXtreme Programming) (página 2) - Monografias.com. [online]. Available from: http://www.monografias.com/trabajos51/programacion-extrema/programacion-extrema2.shtml

44. Desarrollo de software. Tarjetas CRC. *Jummp* [online]. Available from: https://jummp.wordpress.com/2012/01/10/desarrollo-de-software-tarjetas-crc/

45. FAENCARO. Visión General de los Diagramas de Despliegue. *UML* [online]. viernes, de abril de 2013. Available from: http://umldiagramadespliegue.blogspot.com/

46. ¿Qué es https? Definición, concepto y significado. *Quees.la* [online]. Available from: http://quees.la/https/

47. KIOSKEA.NET. Protocolos de mensajería (SMTP, POP3 e IMAP4). [online]. Available from: http://es.kioskea.net/contents/279-protocolos-de-mensajeria-smtp-pop3-e-imap4

48. Arquitectura Software. [online]. Available from: http://www.mastermagazine.info/termino/3916.php

49. Arquitectura cliente-servidor. *DesarrolloWeb.com* [online]. Available from: http://www.desarrolloweb.com/articulos/arquitectura-cliente-servidor.html

50. Patrones de diseño. *Kioskea* [online]. Available from: http://es.kioskea.net/contents/224-patrones-de-diseno

51. BRUCE ECKEL. Pensar en C++. [online]. Available from: http://arco.esi.uclm.es/~david.villa/pensar_en_C++/vol2/index.html

52. LOZU, The. PATRONES DE DISEÑO: DECORATOR (PATRÓN DE DISEÑO). *PATRONES DE DISEÑO* [online]. 7 July 2013. Available from: http://thelozu.blogspot.com/2013/07/decorator-patron-de-diseno.html

53. Patrones de Diseño – Façade | Lycka Bonita. [online]. Available from: http://www.hachisvertas.net/blog/01/2008/11/12/832

54. MARCELLO VISCONTI. *Fundamentos de Ingeniería de Software* [online]. [no date]. Available from: http://www.inf.utfsm.cl/~visconti/ili236/Documentos/08-Patrones.pdf

55. CARMONA, Juan García. Juan García Carmona: GRASP: Experto en información. *Juan García Carmona* [online]. 7 September 2012. Available from: http://juan-garcia-carmona.blogspot.com/2012/09/grasp-experto-en-informacion.html

56. CARMONA, Juan García. Juan García Carmona: GRASP: Creador. *Juan García Carmona* [online]. 7 September 2012. Available from: http://juan-garcia-carmona.blogspot.com/2012/09/grasp-creador.html

57. CARMONA, Juan García. Juan García Carmona: GRASP: Controlador. *Juan García Carmona* [online]. 7 September 2012. Available from: http://juan-garcia-carmona.blogspot.com/2012/09/grasp-controlador.html

58. CARMONA, Juan García. Juan García Carmona: GRASP: Alta cohesión y bajo acoplamiento. *Juan García Carmona* [online]. 7 September 2012. Available from: http://juan-garcia-carmona.blogspot.com/2012/09/grasp-alta-cohesion-y-bajo-acoplamiento.html

59. MVC y su importancia en la Web. *michelletorres.mx* [online]. Available from: http://michelletorres.mx/mvc-y-su-importancia-en-la-web/

60. 2.1. El patrón MVC (Symfony 1.2, la guía definitiva). [online]. Available from: http://librosweb.es/libro/symfony_1_2/capitulo_2/el_patron_mvc.html

61. SOSA, Ivan Hdez. institutotecnologicodehujutla ihs: 2.1 Tareas de la ingeniería de requisitos. *institutotecnologicodehujutla ihs* [online]. 11 March 2013. Available from: http://insitutotec.blogspot.com/2013/03/21-tareas-de-la-ingenieria-de-requisitos.html

62. J.J GUTIÉRREZ. *PRUEBAS DEL SISTEMA EN PROGRAMACIÓN EXTREMA* [online]. Available from: http://www.lsi.us.es/~javierj/investigacion_ficheros/PSISEXTREMA.pdf

63. CARLOS BLÉ JURADO. *Diseño Ágil con TDD* [online]. Available from: http://www.carlosble.com/downloads/disenoAgilConTdd_ebook.pdf

64. ISABEL BLANK. *Pruebas de Funcionalidad.* [online]. March 2015. Available from: http://carolina.terna.net/ingsw3/datos/Pruebas_Funcionales.pdf