UNIVERSIDAD CENTRAL ¨MARTA ABREU¨ DE LAS VILLAS

FACULTAD DE MATEMATICA-FISICA-COMPUTACION

APLICACIÓN INFORMÁTICA PARA LA GESTIÓN DE UN SERVICIO DE DIRECTORIO DE PERSONAS.

FACULTAD DE MATEMÁTICA-FÍSICA-COMPUTACIÓN

Trabajo de Diploma para optar por el título de Ingeniero Informático

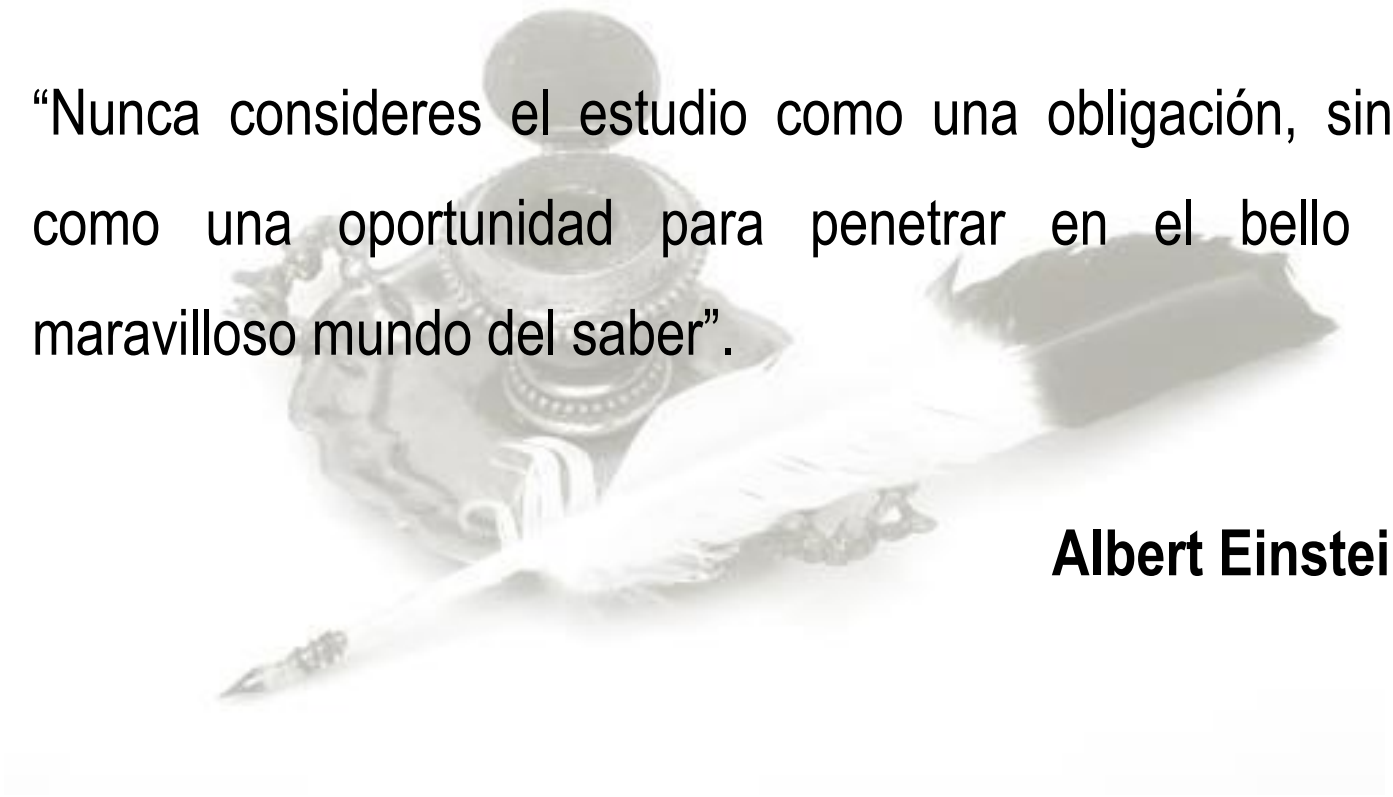
Autor:

Manuel Alejandro Padrón del Pico

Tutor:

Ing. Ernesto Miguel Rodríguez Rodríguez.

Cuidad de Santa Clara, 28 de junio de 2013.
"Año 55 de la Revolución"
2013



“Nunca consideres el estudio como una obligación, sino como una oportunidad para penetrar en el bello y maravilloso mundo del saber”.

Albert Einstein

Datos de Contacto

Declaración de autoría

Declaro ser autor de la presente tesis y reconocemos a la Universidad Central ¨Martha Abreu¨ de las Villas, los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los 28 días del mes de junio del año 2013.

Manuel Alejandro Padrón Del Pico

_____

(Firma del Autor)

Ing. Ernesto Miguel Rodríguez Rodríguez.

_____

(Firma de la tutora)

Agradecimientos

A la Revolución y a nuestra Universidad.

A todos los profesores que aportaron su granito de arena para mi formación profesional.

A toda mi familia por siempre confiar en mí y darme todo su apoyo.

A mi papá por sus consejos y dedicación y sin escatimar esfuerzo alguno, ha sacrificado gran parte de su vida para formarme. Gracias por todo.

A mi hermano Javier por ser mi punto de inspiración, y nunca alcanzarán las palabras para expresarle cuanto lo quiero, que en mi encontrara siempre una mano amigo. Te quiero Macho.

A mi mami por su apoyo y preocupación. Por brindarme tanto cariño y ser tan especial. Por ser la autora intelectual de este resultado, de no haber sido por su insistencia hoy no estaría aquí. Gracias por lo que soy, gracias por todo.

A Vitico, por su apoyo incondicional y siempre estar ahí.

A Belkis por su dedicación y por tantas molestias.

A mis abuelas Milagros y Angélica, a mis tías y mis tíos, a mis primos y en especial a Albertico.

A mi vecinos que estuvieron pendientes de mí todo el tiempo, a mi madrina Enit y a Ovi por su apoyo, a todos gracias.

A mis compañeros de aula que han llenado de satisfacción y alegría estos años de universidad, por estar en las buenas y en las malas, por ser como mi familia.

A mi tutor por su apoyo y ser tan paciente.

A todos, mil gracias.

Dedicatoria

A mis padres y a mi hermano por ser tan especiales y haberme guido hasta aquí, porque sé que hoy están orgullosos de mí. Nuestro sueño ya es realidad.

Resumen

Resumen

El encontrar la solución más acertada entre un conjunto de alternativas factibles siempre ha sido una difícil tarea para el hombre. El uso de los servicios de directorio brinda la posibilidad de encontrar dicha solución. Con este tipo de Servicio de Directorio de Personas (SDP) está arraigado al manejo de grandes cantidades de proceso de datos lo que está enraizado al manejo de amplias alternativas lo que hace que el proceso de aplicación sea complejo y susceptibles a errores.

La idea surge por la necesidad que existe en la MFC de mantener un control y ubicación detallada del personal, y así garantizar el acceso a la información con el objetivo de gestionar cualquier tipo de pesquisa o de obtenerla. Dichos datos que se obtengan pueda usarse para identificar, contactar o localizar a una o varias persona en específico **(Pico, 2012)**. En el presente documento se describen dichas funcionalidades utilizando XP[1] como metodología ágil de desarrollo. Se realiza una descripción de las herramientas y tecnologías utilizadas para el desarrollo de las funcionalidades de la aplicación, además se aplican las pruebas definidas determinándose el correcto funcionamiento del sistema. El resultado de esta investigación, es un servicio que facilita el trabajo a la hora de usar el SDP para resolver problemas de ubicación de las personas.

Palabras Clave: servicio de directorio de personas, gestionar información del personal, identificar, contactar, localizar.

[1] XP es una metodología ágil, que brinda una estrategia desde el punto de vista tecnológico al introducir procedimientos ágiles que permiten actualizar los procesos de software**(Beck, 1996)**.

Índice

ÍNDICE

Introducción 10
Capítulo I. Marco Teórico Referencial 16
1.1 Servicio de directorio 16
1.2 Aplicación Web 16
1.3 Infraestructura digital 16
1.4 Metodologías de Desarrollo de Software 17
1.5 Lenguaje de Modelado 18
1.6 Herramientas Case 18
1.6.1 Visual Paradigm 18
1.7 Lenguajes utilizados 19
1.7.1 Lenguaje de programación del lado del servidor (PHP 5) 19
1.7.2 Lenguaje de programación del lado del cliente (HTML, CSS, JavaScript) 20
1.8 Entornos de Desarrollo Integrado (IDEs) 20
1.8.1 NetBeans 7.2 PHP 20
1.9 Gestores de Base de Datos 21
1.9.1. PostgresSQL 21
1.10 Servidor de Aplicaciones 21
1.10.1 Servidor Web (XAMPP v.1.8) 21
1.11 Framework de desarrollo 22
1.11.1 Propel 22
Conclusiones Parciales 23
Capítulo 2. Características del Sistema 24
2.1 Descripción del problema 24
2.2 Solución Propuesta 24
2.3 Concepción inicial del sistema 25
2.4 Captura de requisitos 25
2.4.1 Selección de una lista de datos de personas (LDP) 26
2.5 Diseño de metáfora 28
2.6 Roles de usuario 28
2.7 Historias de Usuario (Hurtado) 28
2.8 Plan de Iteración 32
2.8.1 Plan de duración de las iteraciones 32
2.9 Plan de entrega 33
2.10 Fases de Diseño 33
2.11 Tarjetas CRC 34
2.12 Estilo Arquitectónico Modelo- Vista- Controlador (MVC) 37
2.13 Patrones de diseño 37
2.13.1 Patrones GRASP (Patrones de Software para la Asignación General de Responsabilidad) 38
2.14 Modelo de Datos 39

Índice

Conclusiones Parciales 41
Capítulo 3 Implementación y Prueba 42
3.1 Implementación 42
3.3.1 Diagrama de componentes 42
3.2 Desarrollo de las iteraciones 43
3.3 Estándares de codificación 45
3.4 Pruebas de aceptación o caso de prueba 46
3.4.1 Diseño de los casos de prueba 46
3.5 Diseño de los casos de prueba 47
Conclusiones Parciales. 47
Conclusiones Generales. 49
Recomendaciones 50
Referencias Bibliográficas 51
Bibliografías 52
Anexos 53
Anexo 1 Plantilla de Concepción Inicial del Sistema 53
Anexo 2 Descripción del Modelo de Datos 58
Anexo 3 Tareas de Ingenierías 59
Anexo 4 Estándares de codificación 63
Anexo 5 Caso de Pruebas de Aceptación 68
Anexo 6 Interfaces de Usuario 70

ÍNDICE DE FIGURAS

Figura 1: Esquema de la metodología XP. 18
Figura 2: Diagrama de Clases 36
Figura 3: Modelo Vista Controlador 37
Figura 4: Modelo de Datos 40
Figura 5: Diagrama de componentes – Búsqueda Avanzada 43
Figura 6: Diagrama de componentes – Gestionar Usuario 43
Figura 7: Esquema caja negra 46
Figura 8: Declaración por línea 63
Figura 9: Sentencia SetEstudiante 64
Figura 10: Sentencia Compuestas 65
Figura 11: Sentencia if – else 66
Figura 12: Sentencia for 67
Figura 13: Método 67
Figura 14: Variables 68
Figura 15. Introducir criterio de búsqueda. 70
Figura 16: Autentificación del administrador. 71
Figura 17: Agregar persona 71
Figura 18: Eliminar persona 72

Figura 19: Cartel de alerta. 72

ÍNDICE DE TABLAS

Tabla 1: Lista de datos 28
Tabla 2: Roles de Usuario 28
Tabla3:HU_1. Gestionar criterio de búsqueda. 29
Tabla 4:HU_2. Generar criterio. 29
Tabla 5:HU_3. Autenticar usuario 30
Tabla 6:HU_4. Gestionar usuario 31
Tabla 7: Puntos Estimación 31
Tabla 8: Plan de duración de las iteraciones 33
Tabla 9: Plan de entrega 33
Tabla 10: Tarjeta CRC 1 34
Tabla 11: Tarjeta CRC 2 34
Tabla 12: Tarjeta CRC 3 35
Tabla 13: Tarjeta CRC 4 35
Tabla 14: Tarjeta CRC 5 35
Tabla 15: Tarjeta CRC 6 35
Tabla 16: Tabla Decisión de la Base de Datos 40
Tabla 17: Tarea de Ingeniería 1 44
Tabla 18: Tarea de Ingeniería 2 44
Tabla 19: Tarea de Ingeniería 3 45
Tabla 20: Diseño de Caso de Prueba de Aceptación 1 47
Tabla 21: Tabla de Roles 56
Tabla 22: Tablas de la Base de Datos 59
Tabla 23: Tarea de Ingeniería 1 59
Tabla 24: Tarea de Ingeniería 2 60
Tabla 25: Tarea de Ingeniería 3 60
Tabla 26: Tarea de Ingeniería 4 61
Tabla 27: Tarea de Ingeniería 5 61
Tabla 28: Tarea de Ingeniería 6 61
Tabla 29: Tarea de Ingeniería 7 62
Tabla 30: Tarea de Ingeniería 8 62
Tabla 31: Tarea de Ingeniería 9 63
Tabla 32: Tarea de Ingeniería10 63
Tabla 33: Caso de prueba de Aceptación 1 68
Tabla 34: Caso de prueba de Aceptación 2 69
Tabla 35: Caso de prueba de Aceptación 3 69

Introducción

Durante la historia nos hemos dado cuenta del papel protagónico que ha jugado la comunicación en el desarrollo de la sociedad y no solo el que jugó sino el que viene desempeñando actualmente, donde no existen obstáculos, como por ejemplo, una persona en Asia pueda hablar en tiempo real con otro en América, donde millones y millones de personas se conectan a internet y transfieren información sin importar el lugar donde se encuentren, por lo tanto el poder de la información que usted adquiera ejerce una gran flujo de datos.

Para tomar una buena decisión es importante realizarnos la siguiente pregunta, ¿Qué tenemos que hacer para obtener información de la forma más rápida posible? Existen casos que para lograr una información es muy difícil, ya que la misma a veces resulta incapaz de lograrla porque no la tiene nadie o no hay demasiada para alcanzarla, de esto es que va la respuesta, ya que la manera más rápida de acceder a la información es conocer dónde encontrarla y para esto se debería almacenar y centralizar la inquisición para que otros la puedan reutilizar.

Para darle solución a esta problemática el hombre tuvo muchas ideas y la más importante fue la creación de bibliotecas donde se almacenaban escrituras y libros de generación en generación, de esta forma la información se fue transmitiendo y nunca se perdió. En la actualidad esa idea no solo se mantiene, sino que también la han utilizado para la creación de herramientas informáticas donde se almacena dicha información, sino también de organizarlas y alcanzarlas.

Basándome en lo antes expuesto, este trabajo se fundamenta especialmente en implementar un software especializado para el almacenamiento de información pública de personas en la UCLV y éste sería un servicio de directorios de personas, por lo que tendrá como principal tarea plasmar la información púbica de todo el personal que se relacione de forma directa con la universidad, dígase estudiantes, profesores y trabajadores vinculados a ésta.

La idea surge por la necesidad que existe en la MFC de logar un control más detallado del personal que pertenece a ésta y garantizará el acceso a la información por los usuarios autorizados que se encargaran de su actualización; además de gestionar, que los dispositivos o mecanismos utilizados para su seguridad no impedirán obtener los datos deseados en un momento dado. Esto garantizará además, tener un control de toda la información del personal para su uso con el objetivo de gestionar algún tipo de

información o de obtenerla refiriéndose a la documentación que pueda usarse para identificar, contactar o localizar a una persona en específico, junto a otras fuentes de información para hacerlo **(Rodríguez, 2012)**.

El servicio de directorio de persona (SDP) tendrá todas aquellas medidas preventivas y reactivas del hombre, de las organizaciones y de los sistemas tecnológicos que permitan resguardar y proteger la información buscando mantener **(Borge, 2007)**:

- ✓ La *confidencialidad* de acceso y control de la información únicamente por personas que cuenten con la debida autorización.
- ✓ La *disponibilidad* de la información de encontrarse a disposición de quienes deben acceder a ella, ya sean personas, procesos o aplicaciones.
- ✓ La *Integridad* de la misma, a la propiedad que busca mantener los datos libres de modificaciones no autorizadas.

Por su parte la organización de estos tipos de datos, actualmente es un factor de éxito en las empresas. La entrada de datos digitales revolucionó por completo el sector de los servicio de información. Donde se desarrollaron las bases de datos y aplicaciones dedicadas a optimizar su gestión, debido a la relevancia de este aspecto en el mundo de la información.

La gestión de la información personal es la práctica y el estudio de las actividades que las personas desempeñan con el fin de organizar y utilizar la información que obtienen para resolver necesidades cotidianas. Tales actividades aspiran a alcanzar tres objetivos **(Borge, 2007)**:

- ✓ Obtener, organizar y reencontrar la información necesaria para completar tareas y cumplir con deberes diversos.
- ✓ Garantizar que un documento o un canal de gestión de datos, una vez localizado, volverá a estar disponible cuando se necesite.
- ✓ Aprovechar al máximo los recursos personales para aumentar la productividad y mejorar la calidad de vida.

Una buena organización garantiza disponer de la información precisa al instante, haciéndola visible sólo a aquellos usuarios que la necesiten. Además, integrar una excelente estructura permite una gran

coordinación y comunicación entre los miembros. Otros beneficios de este sistema de directorio, será el de reducir el esfuerzo, al tener disponible efectivamente la comunicación entre las personas.

La comunicación es esencial ya que mejora los costos de operación o se reducen, son amigables al usuario, la probabilidad de que una falla de información afecte al sistema es baja, ya que existirá una autonomía e independencia entre los datos, aun cuando el servicio de directorio de usuario se comporta ante el cliente como un todo integrado **(Autores, 2009)**.

En esencia este tipo de servicio de directorio ofrece al usuario un modo seguro y confiable para la gestión de datos que servirá como medio para ubicar y obtener a la persona que desea saber, lo que suele ser difícil o engorroso.

La información puede llegar en mal momento, logra estar en un lugar inoportuno, consigue perderse en un laberinto o alcanza caer en el olvido. Además, suele estar fragmentada en diversas formas, cada una de las cuales implica un sistema de organización y un uso distintos. La gestión de información personal estudia qué conviene hacer para resolver esos problemas y lograr que esté al servicio de la persona en su quehacer cotidiano.

En fin, este tipo de información es un recurso valioso, aunque el valor no le es intrínseco. Es un recurso muy abundante y por ello conviene organizarla, porque gestionar información personal es la manera más tangible de gestionar otras tácticas: tiempo, atención y conocimiento.

El modo para localizarlos se describe en las diferentes categorías:

- ✓ Trabajador Docente: Decana (o), Vicedecana (o), Secretaria (o) y Profesores.
- ✓ Trabajador u Obrero: Técnicos, Auxiliares Generales, Bibliotecarias (o), etc.
- ✓ Personal Estudiantil: Todos los estudiantes de la facultad.

Es nuestro fin de no incluir datos personales no inconvenientes por las personas incluidas en el servicio de directorio, con el fin de limitar cualquier tipo de uso no deseado.

El servicio de directorio de personas pretende ser transparentes en su funcionamiento, los usuarios deben saber cómo consiguen y clasifican la información.

Es una clave tener conocimiento sobre la creación de nuestra identidad, eso es lo que hay que tener en cuenta, tanto para nuestros usuarios como para otras personas, ya que estas dependen de un mundo real y de una información confiable de este tipo de perfiles online.

De la situación anteriormente planteada, se formula el siguiente **problema de investigación**: ¿Cómo gestionar las características de las personas en la facultad de Matemática, Física y Computación?

A partir del problema planteado se define como **objeto de estudio**: La gestión de un sistema de servicio de directorio de personas.
Para solucionar el problema planteado se define como **objetivo general**: Desarrollar una aplicación informática que facilite la ubicación y caracterización de las personas.

Para cumplir este objetivo se desglosan los siguientes **objetivos específicos**:

- ✓ Establecer el marco teórico referencial sobre el servicio de directorio de personas, y caracterizar los métodos para la gestión de datos SDP.
- ✓ Caracterizar el entorno de desarrollo de la aplicación.
- ✓ Modelar las funcionalidades de los métodos como parte del sistema al servicio de directorio.
- ✓ Implementar las funcionalidades modeladas.
- ✓ Validar la aplicación propuesta.

Para dar cumplimiento a los objetivos específicos anteriores se trazaron las siguientes tareas de investigación:

- ✓ Elaborar el marco conceptual de la investigación.
- ✓ Estudiar los métodos para un servicio de directorio.
- ✓ Estudiar la metodología definida para el desarrollo del sistema.
- ✓ Estudiar patrones de diseño.
- ✓ Estudiar las herramientas y tecnologías a utilizar para el desarrollo de la propuesta de solución.
- ✓ Definir las funcionalidades del sistema.
- ✓ Definir mecanismos de diseño a utilizar.
- ✓ Realizar diagramas de clases del diseño.
- ✓ Describir las clases del diseño.
- ✓ Realizar modelo de datos.

- ✓ Realizar diagrama de componentes.
- ✓ Definir estándares de codificación.
- ✓ Realizar descripción por funcionalidades.
- ✓ Implementar las funcionalidades definidas.
- ✓ Validar el software mediante la aplicación de pruebas de caja negra.

Se plantea la siguiente **idea a defender** para la investigación: Si se desarrolla el servicio de directorio de personas se logrará eliminar la complejidad de la confección manual de los mismos y se evitará que el proceso sea susceptible a errores.

Se utilizaron los siguientes métodos de investigación:

Método teórico: Permite descubrir en el objeto de investigación las relaciones esenciales y las cualidades fundamentales, no detectables de maneras senso-perceptual. Por ello se apoya básicamente en los procesos de abstracción, análisis, síntesis, inducción y deducción, por lo que se utilizó **(R., En línea)**:

- ✓ **Analítico-sintético: E**n esta investigación se emplea este método para analizar las teorías, documentos y todo tipo de información que se tiene sobre el desarrollo del software; permitiendo la extracción de los elementos más importantes que se relacionan con el objeto de estudio y lograr así una mejor comprensión del mismo. También se utiliza para el análisis de los resultados más significativos que se han obtenido actualmente en ese campo **(Hurtado, 2009)**.
- ✓ **Histórico-lógico:** A través de este método se logró un mayor entendimiento y conocimiento de las diferentes técnicas y herramientas de modelado a utilizar, dando la posibilidad de analizar la trayectoria histórica de los mismos, así como su evolución y desarrollo.
- ✓ **Modelación:** Este método es de gran importancia para la investigación debido a que brinda la posibilidad de representar las propiedades y funcionalidades que tendrá el sistema a desarrollar.

Métodos empíricos: Estos métodos posibilitan revelar las relaciones esenciales y las características fundamentales del objeto de estudio, accesibles a la detección de la percepción, a través de procedimientos prácticos con el objeto y diversos medios de estudio, y se utilizó **(R., En línea)**:

- ✓ **Observación:** Con este método se logró recopilar información acerca de los principales conceptos y características de los servicios de directorios **(Rodriguez, En linea).**

- ✓ **Análisis de documentos:** Se estudiaron los documentos relacionados con la metodología y los métodos a implementar.

Técnicas de recopilación de información: La recopilación de datos se utiliza para verificar los métodos empleados en lo investigado, para llegar a la conclusión del suceso, teniendo las pruebas y una serie de pasos que se llevaron a cabo para comprobar la hipótesis planteada, como son:

- ✓ **Entrevistas**: Es una técnica para obtener datos, que consiste en un diálogo entre dos personas. En este caso se empleó con el fin de obtener información referente a las herramientas a emplear, la metodología a implementar, además de obtener datos referentes a la experiencia de personas que han utilizado los diferentes servicios de directorios.

El documento está estructurado en 3 capítulos:

Capítulo 1 "Fundamentación teórica de la propuesta a desarrollar":

En este capítulo se hace breve descripción del estudio realizado que sirvió de base para llevar a cabo la investigación. Adicionalmente fue realizado un estudio de las principales tecnologías y herramientas a utilizar en el desarrollo de la aplicación Web.

Capítulo 2 "Características del sistema a desarrollar":

En este capítulo se describen las características básicas y fundamentales del sistema informático a desarrollar. En el mismo, se realiza una breve descripción del problema en cuestión, especificándose los requisitos funcionales que plantea el cliente y que a su vez el sistema debe cumplir.

Se da a conocer todo lo referente al modelado del negocio, incluyendo las historias de usuario del sistema y una breve descripción de cada uno de ellos, englobando todo lo referente al modelado del diseño específicamente.

Capítulo 3 "Implementación y Pruebas del Sistema":

En este capítulo se realizan los diagramas de despliegue y componentes como resultado de la implementación del sistema, además de las pruebas realizadas una vez concluido el desarrollo de la aplicación.

Capítulo I. Marco Teórico Referencial

Para el desarrollo de la presente investigación se hizo la revisión, estudio y análisis de los diversos temas, que permitieron establecer el marco teórico referencial, para la obtención de los resultados que dan cumplimiento a los objetos propuestos. Los principales temas abordados están relacionados con el objeto de estudio definido en el diseño de investigación.

En el presente capitulo se muestra el análisis realizado, a partir del estudio sobre los diferentes servicios de directorio que están en funcionamiento en la actualidad. Por último, para el desarrollo de la aplicación es necesario definir las tecnologías y herramientas a utilizar.

1.1 Servicio de directorio

Los directorios representan contenedores de información descriptiva basada en atributos, apoyado en sofisticadas capacidades de filtrado. Los directorios por lo general no admiten transacciones complicadas como las que se encuentran en los sistemas de bases de datos diseñados para manejar grandes y complejos volúmenes de actualizaciones. Las actualizaciones de los directorios son normalmente cambios simples, o todo o nada, siempre y cuando estén permitidos. Es decir es una base de datos específicamente diseñada para la búsqueda y navegación **(Carter, 2011)**.

1.2 Aplicación Web

Son aquellas aplicaciones que los usuarios pueden utilizar accediendo a un servidor web a través de Internet o de una intranet mediante un navegador. En otras palabras, es una aplicación software que se codifica en un lenguaje soportado por los navegadores web en la que se confía la ejecución al navegador.
Las aplicaciones web son notorias debido a lo práctico del navegador web como cliente ligero, a la independencia del sistema operativo, así como a la facilidad para actualizar y mantener aplicaciones web sin distribuir e instalar software a miles de usuarios potenciales.

1.3 Infraestructura digital

En el desarrollo de software, una infraestructura digital, es una estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos de software concretos, con base a la cual otro

proyecto de software puede ser más fácilmente organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas, y un lenguaje interpretado, entre otras herramientas, para así ayudar a desarrollar y unir los diferentes componentes de un proyecto.

1.4 Metodologías de Desarrollo de Software

Metodología de desarrollo de software en ingeniería de software, es un marco de trabajo usado para estructurar, planificar y controlar el proceso de desarrollo en sistemas de información. De las que se encuentra XP.

La programación extrema o eXtreme Programming (XP) es un enfoque de la ingeniería de software. La programación extrema se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad. Los defensores de XP consideran que los cambios de requisitos sobre la marcha son un aspecto natural, inevitable e incluso deseable del desarrollo de proyectos. Creen que ser capaz de adaptarse a los cambios de requisitos en cualquier punto de la vida del proyecto es una aproximación mejor y más realista que intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos después en controlar los cambios en los requisitos. Se puede considerar la programación extrema como la adopción de las mejores metodologías de desarrollo de acuerdo a lo que se pretende llevar a cabo con el proyecto, y aplicarlo de manera dinámica durante el ciclo de vida del software **(Beck, 1996)**.

Las pruebas han de garantizar que en la refactorización no se ha introducido ningún fallo. Este tipo de metodología nos provee de una mayor simplicidad en el código ya que es la mejor manera de que las cosas funcionen. Cuando todo funcione se podrá añadir funcionalidad si es necesario.

La programación extrema apuesta que es más sencillo hacer algo simple y tener un poco de trabajo extra para cambiarlo si se requiere, que realizar algo complicado y quizás nunca utilizarlo. La simplicidad y la comunicación son extraordinariamente complementarias. Con más comunicación resulta más fácil identificar qué se debe y qué no se debe hacer. Mientras más simple es el sistema, menos tendrá que comunicar sobre este, lo que lleva a una comunicación más completa, especialmente si se puede reducir el equipo de programadores.

Figura 1: Esquema de la metodología XP.

1.5 Lenguaje de Modelado

El Lenguaje Unificado de Modelado (UML) es un lenguaje visual para especificar, construir y documentar los artefactos de sistemas intensivos en software. **(IBM, IBM, 2011)** Prescribe un conjunto de notaciones y diagramas estándar para modelar sistemas orientados a objetos, y describe la semántica esencial de lo que estos diagramas y símbolos significan. Mientras que ha habido muchas notaciones y métodos usados para el diseño orientado a objetos, ahora los modeladores sólo tienen que aprender una única notación. UML no es un método de desarrollo, lo que significa que no sirve para determinar qué hacer en primer lugar o cómo diseñar el sistema, sino que simplemente le ayuda a visualizar el diseño y a hacerlo más accesible para otros. UML está controlado por el grupo de administración de objetos (OMG) y es el estándar de descripción de esquemas de software. **(2011, línea, 2011)**

1.6 Herramientas Case

1.6.1 Visual Paradigm

Visual Paradigm es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. También proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML. Presenta licencia gratuita y comercial. Es fácil de instalar y actualizar y compatible entre ediciones. Generación de bases de datos, transformación de diagramas de Entidad-Relación en tablas de base de datos; Importación y exportación de ficheros XMI **(F., 1998)**. .

1.7 Lenguajes utilizados

1.7.1 Lenguaje de programación del lado del servidor (PHP 5)

PHP (siglas en inglés Hypertext Preprocessor), es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas. El rendimiento es muy bueno y verdaderamente eficiente, utilizando un servidor modesto puedes atender millones de peticiones al día. PHP es *gratuito, además* es *OpenSource* es decir que se tiene acceso al código fuente. Este tipo de lenguaje es completamente expandible ya que puede interactuar con muchos motores de base de datos tales como MySQL, Oracle, PostgreSQL, entre otros. Cuenta con cuatro grandes características que son imprescindibles para su huso **(Ullman, 2007)**:

- ✓ **Velocidad:** No solo la velocidad de ejecución, la cual es importante, sino además no crear demoras en la máquina. Por esta razón no debe requerir demasiados recursos de sistema. PHP se integra muy bien junto a otro software, especialmente bajo ambientes Unix.
- ✓ **Estabilidad:** PHP utiliza su propio sistema de administración de recursos y dispone de un sofisticado método de manejo de variables, conformando un sistema robusto y estable.
- ✓ **Seguridad:** Provee diferentes niveles de seguridad, estos pueden ser configurados desde el archivo .ini.

PHP dispone de una amplia gama de librerías, y agregarle extensiones es muy fácil. Esto le permite al PHP ser utilizado en muchas áreas diferentes, tales como encriptado, gráficos, XML y otras.

1.7.2 Lenguaje de programación del lado del cliente (HTML, CSS, JavaScript)

El lenguaje de marcado de hipertexto (HTML) es predominante para la elaboración de páginas web. Además de ser usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes. También puede describir la apariencia de un documento, y puede incluir un scriptel cual puede afectar el comportamiento de navegadores web y otros procesadores de HTML **(Raymond, Citado 2012)**, por ejemplo JavaScript.

JavaScript es un lenguaje de programación interpretado. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico. Se utiliza principalmente en su forma del lado del cliente (client-side), implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas, en bases de datos locales al navegador **(Crockford, Citado 2012)**.

Un formato específico puede ser aplicado al texto por medio de hojas de estilo en cascada (CSS), es un lenguaje usado para definir la presentación de un documento estructurado escrito en HTML o XML. La idea que se encuentra detrás del desarrollo de CSS es separar la estructura de un documento de su presentación. Al utilizar CCS hay un control centralizado de la presentación de un sitio web completo con lo que se agiliza de forma considerable la actualización del mismo. Por lo que el lenguaje de programación en páginas web es de gran valor porque ayuda la accesibilidad del documento, optimiza el ancho de banda de la conexión y estas aplicaciones se desenvuelven en un ambiente ágil y dinámico **(Consortium, 20-06-2009)**.

1.8 Entornos de Desarrollo Integrado (IDEs)

1.8.1 NetBeans 7.2 PHP

NetBeans IDE para PHP 7.2 ofrece una serie de características específicas para el desarrollo con PHP 5.4. Presenta auto completamiento de código PHP así como soporta estilos matrices. NetBeans también reconoce los rasgos y las variables anónimas de objeto, introduce soporte para la integración continua. Puede utilizarse plantillas para Jenkins que son empleados para proyectos PHP.

El IDE NetBeans es totalmente compatible con el desarrollo iterativo, lo que las pruebas de los proyectos PHP siguen los patrones para el desarrollo web. Además soporta muchos frameworks de desarrollo como Symfony, por lo que fue escogido para el desarrollo de la aplicación. El IDE proporciona autocompletado

de código y reconoce la sintaxis de estos marcos, así como los marcos originales de Symfony. NetBeans proporciona soporte básico para FTP / SFTP, suficiente para un desarrollador independiente para trabajar en un proyecto simple **(Dorado, Novienbre, 2005)**.

1.9 Gestores de Base de Datos

1.9.1. PostgresSQL

PostgresSQL es un sistema de gestión de base de datos relacional orientada a objetos y libre, publicado bajo la licencia de Distribución de Software Berkeley (BSD, Berkeley Software Distribution, según sus siglas en inglés). PostgresSQL implementa el uso de retrocesos, subconsultas y transacciones, haciendo su funcionamiento mucho más eficaz. Posee la capacidad de comprobar la integridad referencial, así como también la de almacenar procedimientos en la propia base de datos. Tiene mejor soporte para disparadores y procedimientos en el servidor **(test, 13-01-2011)**.

1.10 Servidor de Aplicaciones

1.10.1 Servidor Web (XAMPP v.1.8)

Un servidor web o servidor HTTP 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.
XAMPP, es un servidor de plataforma libre, es un software que integra en una sola aplicación, un servidor web Apache, intérpretes de lenguaje de scripts PHP, un servidor de base de datos MySQL, un servidor de FTP FileZilla, el popular administrador de base de datos escrito en PHP, MySQL, entre otros módulos.
XAMPP es una herramienta de desarrollo que te permite probar tu trabajo (páginas web o programación por ejemplo) en tu propio ordenador sin necesidad de tener que acceder a internet. La mayor ventaja de XAMPP es que es muy fácil de instalar y las configuraciones son mínimas o inexistentes, lo cual nos ahorra bastante tiempo **(XAMPP, Febrero, 2012)**.

1.11 Framework de desarrollo

Un frameworks un conjunto de herramientas, librerías y procesos desarrollados con el único fin de aminorar el trabajo del desarrollador. Symfony, un completo framework diseñado para optimizar el desarrollo de las aplicaciones web. Para empezar, 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 se diseñó para que se ajustara a los siguientes requisitos **(Nuñez, 30-07-2009)**:

- ✓ Fácil de instalar y configurar en la mayoría de plataformas (y con la garantía de que funciona correctamente en los sistemas Windows, Unix y Linux).
- ✓ Independiente del sistema gestor de bases de datos. Su capa de abstracción y el uso del ORM Propel, permiten cambiar con facilidad de SGBD en cualquier fase del proyecto.
- ✓ Utiliza programación orientada a objetos, de ahí que sea imprescindible PHP 5.
- ✓ Aunque utiliza estilo arquitectónico MVC (Modelo vista controlador), tiene su propia forma de trabajo en este punto, con variantes del MVC clásico como la capa de abstracción de base de datos, el controlador frontal y las acciones.
- ✓ Sigue la mayoría de mejores prácticas y patrones de diseño para la web.
- ✓ Código fácil de leer que incluye comentarios de phpDocumentor y que permite un mantenimiento muy sencillo.
- ✓ Fácil de extender, lo que permite su integración con las librerías de otros fabricantes.

Symfony se considera un framework elegante, estable, productivo y muy bien documentado. Por este motivo se decidió apostar por Symfony utilizándolo en nuestra aplicación.

1.11.1 Propel

Es un mapeo objeto-relacional (ORM) de código abierto escrito en PHP. Es además una parte integral del framework Symfony. La función primaria de Propel es proveer un mapa entre las clases de PHP y tablas de bases de datos. Para lograr esto, Propel incluye un componente generador que usa generación de

código fuente para construir clases PHP basadas en una definición de modelo de datos (datamodel definition) escrita en XML (Lenguaje de marcas extensibles). Propel también incluye un componente de ejecución que maneja conexiones, transacciones y cualquier regla de idiosincrasia que describa el funcionamiento del sistema de gestión de bases de datos relacionales que este siendo usado con Propel **(Nuñez, 30-07-2009)**.

Conclusiones Parciales

- ✓ Se abordaron los principales conceptos relacionados con la investigación.
- ✓ Además el estudio de los servicios de directorio nos posibilita la selección de las herramientas y tecnologías que permitirán su gestión.
- ✓ Se desarrollará una aplicación Web donde se utilizará XP como metodología de desarrollo de software, que minimizará el tiempo de implementación de la aplicación y como lenguaje de programación PHP.
- ✓ Dado que la información que se necesita para el proyecto se ha utilizado el framework Symfony 1.4 que permite acceder desde NetBeans.
- ✓ Se seleccionó XAMPP versión 7 como servidor de aplicaciones que se puede integrar con NetBeans 7.2 PHP, como IDE seleccionado.
- ✓ Para la gestión de usuarios y almacenamiento de otras informaciones se seleccionó el gestor de bases de datos PostgreSQL 9.1.
- ✓ Se analizaron los patrones de diseño que posteriormente serán aplicados en la implementación de la aplicación.

Capítulo 2. Características del Sistema

En el presente capitulo se describen los procesos asociados a la gestión de un servicio de sistema de directorio de personas. Se identifican las Historias de los Usuarios (UH) que serán implementados posteriormente y se elaboran las tarjetas CRC. Se define además los requisitos funcionales y no funcionales con el objetivo de lograr que la aplicación sea confiable y fácil de manipular. Con el uso de los patrones de diseño y el estilo arquitectónico se obtiene la estructura de la aplicación que será implementada.

2.1 Descripción del problema

Por las características de un servicio de directorio, es fundamental destacar que u error en el mismo provocaría desconfianza y baja aceptación, en muchas ocasiones la obtención de datos para un tipo de información toma un alto riesgo, tales como equivocaciones de datos y hasta errores susceptibles. En otros casos se ha visto donde una incorrecta información ha provocado algunos deslices para la ubicación o procedencia de las personas. Hoy en día la necesidad de un proceso rápido y confiable en el manejo de gran cantidad de volúmenes de información a tener en cuenta es realizada de forma manual, lo cual trae como consecuencia que el proceso se vea inmenso en fallas. De aquí que la premisa será implementar un servicio de directorio de personas que permita reducir los tiempos e impedir que la misma sea susceptible a errores.

2.2 Solución Propuesta

Actualmente nuestra facultad de Matemática-Física-Computación presenta dificultades para obtener información de los distintas personas que se encuentran ejerciendo una labor por lo que se propone desarrollar una aplicación que permita brindar un servicio de directorio de persona para el conocimiento visualización y obtención de datos que se pueda recopilar de cada uno de sus usuarios, siendo estas las principales funcionalidades que la compone. A continuación se describen brevemente:

- ✓ Buscar usuario: Permite una obtención de datos de los diferentes usuarios que deseas.
- ✓ Visualizar usuario: Posibilita visualizar el usuario buscado con los distintos datos correspondidos.

La aplicación contara además con otras secciones que permitirán obtener la siguiente información:

- ✓ Contactos: Permitirá al usuario si desea hacer una sugerencia para el arreglo o modificación de sus datos, contactar con los encargados del sistema.

La aplicación estará disponible para todos los usuarios que tengan interés en interactuar con la misma. Solo podrá hacer uso de las funcionalidades de la aplicación el usuario encargado y autentificado, el cual tendrá todos los privilegios que le permitirán tener un total manejo sobre la aplicación garantizando su mantenimiento, actualización y correcto funcionamiento.

2.3 Concepción inicial del sistema

En la etapa de concepción del sistema se genera la plantilla de concepción del sistema, en ella se especifican los aspectos generales organizativos, su objetivo, principales involucrados y otros aspectos que permiten la posterior organización del desarrollo de la aplicación. Esta plantilla se muestra en el *Anexo 1*.

2.4 Captura de requisitos

La Ingeniería de Requisitos, es el proceso de desarrollar una especificación de Software. Las especificaciones pretenden comunicar las necesidades del sistema del cliente a los desarrolladores del sistema. Trata de los principios, métodos, técnicas y herramientas que permiten descubrir, documentar y mantener los requisitos para sistemas basados en computadora, de forma sistemática y repetible **(EcuRed, 28-02-2013)**.Para la extracción de los requisitos se emplearon las herramientas y métodos existentes garantizando una correcta redacción y análisis de los mismos.

Luego de realizar entrevistas y posteriormente la discusión con el cliente para determinar los requisitos del sistema en las posibles áreas donde se aplicará la herramienta, se lograron extraer los requisitos. Como resultado de la aplicación de estas técnicas se obtuvo la plantilla de modelos de historias de usuarios del negocio y la lista de reserva del producto, estos elementos son descritos en los siguientes epígrafes. Estos elementos que constituyen los requisitos de la aplicación a implementar fueron validados por el cliente, partiendo de que se describen estos artefactos de conjunto los desarrolladores y el cliente. Luego se sometieron a 3 rondas de revisiones lo cual permitió llegar a un consenso y claridad de las historias de usuario y sus descripciones.

2.4.1 Selección de una lista de datos de personas (LDP)

La lista de datos de personas, es un artefacto generado en la captura de requisitos, donde se describen los mismos como funcionalidades que el sistema debe cumplir en su desarrollo. Para su construcción se cuenta con la colaboración del gerente del proyecto, el cliente y miembros del equipo.

Prioridad	Ítem	Descripción	Estimación	Estimado por	Asignado a
Requisitos Funcionales					
Muy Alta					
	1	Gestionar Criterios de búsqueda	1	programador	Manuel Padrón
Alta					
	2	Generar Criterio	3	programador	Manuel Padrón
Media					
	3	Gestionar usuarios	3	programador	Manuel Padrón
Requisitos No Funcionales					
Usabilidad					
	1	El software tendrá siempre la posibilidad de ayuda disponible para cualquier tipo de usuario, lo que le permitirá un avance considerable en la explotación de la aplicación en todas sus funcionalidades.		programador	Manuel Padrón
	2	Debe poseer una interfaz agradable para el cliente.		Diseñador	Manuel Padrón
Fiabilidad					
	3	El sistema estará disponible 24 horas del día, los siete días de la semana.		programador	Manuel Padrón
	4	La herramienta de implementación a utilizar tiene soporte para recuperación ante fallos y errores.		programador	Manuel Padrón
Eficiencia					
	5	Tiempo de respuesta promedio a las peticiones		programador	Manuel Padrón

		que se realizan al servidor no deberá ser mayor de7 segundos.		
Seguridad				
	6	Protección contra acciones no autorizadas o que puedan afectar la integridad de los datos.	programador	Manuel Padrón
	7	El sistema debe garantizar la confidencialidad, integridad y disponibilidad de la información que se procese en el sistema.	programador	Manuel Padrón
Soporte				
	8	Soporte para grandes volúmenes de datos y velocidad de procesamiento.	programador	Manuel Padrón
	9	Tiempo de respuesta rápido en accesos concurrentes.	programador	Manuel Padrón
Restricciones de diseño				
	10	El lenguaje de programación es PHP.	programador	Manuel Padrón
	11	La herramienta IDE de desarrollo utilizada será NetBeans 7.2 para PHP.	programador	Manuel Padrón
	12	La herramienta case utilizada es Visual Paradigm 8.0 para modelado.	programador	Manuel Padrón
	13	La herramienta gestor de base de datos es el PostgreSQL 9.1	programador	Manuel Padrón
Interfaz				
	14	El sistema tiene que ofrecer una interfaz amigable, fácil de operar.	Diseñador	Manuel Padrón
	15	Diseño sencillo, con pocas entradas, permitiendo que no sea necesario mucho entrenamiento para que los usuarios puedan utilizar el sistema.	Diseñador	Manuel Padrón

Tabla 1: Lista de datos

2.5 Diseño de metáfora

A partir de la definición de las funcionalidades descritas en la lista de datos de personas (LDP) es posible establecer las historias de usuarios, roles y prototipos del sistema y las tareas ingenieriles que permiten su desarrollo siendo estas las actividades que se realizan y se describen a continuación.

2.6 Roles de usuario

Determinar a quién está dirigido un sistema informático es una de las premisas durante el desarrollo de cualquier producto. Para eta aplicación se identificaron los siguientes roles de usuario: el administrador y usuario invitado. A continuación se muestra una tabla con las descripciones de cada una de los roles.

Roles	Descripción
Administrador	Encargado de gestionar toda la información de la aplicación. Establece el control total de la aplicación Web.
Invitado	Toda persona que acceda a la aplicación. Su acceso y nivel de visualización de la información será definido por el administrador.

Tabla 2: Roles de Usuario

2.7 Historias de Usuario (Hurtado)

Teniendo en cuenta cada una de las funcionalidades de la aplicación, se determina por los programadores que las HU deben ser implementadas entre unas y tres semanas dependiendo del riesgo de desarrollo. Se deciden implementar cuatro HU, donde las primeras serán las de nivel de riesgo alto que son además las definidas por el cliente con mayor prioridad de desarrollo y luego se implementarán las HU de riesgo medio **(Penadés).** A continuación se describen las cinco HU.

Historia de Usuario	
Número: HU_1	Nombre: Buscar personas
Usuario: Administrador, Usuario invitado	Iteración Asignada: Estimación 1
Prioridad en Negocio: Alta	Riesgo de desarrollo: Alta
Puntos Estimados: 1	

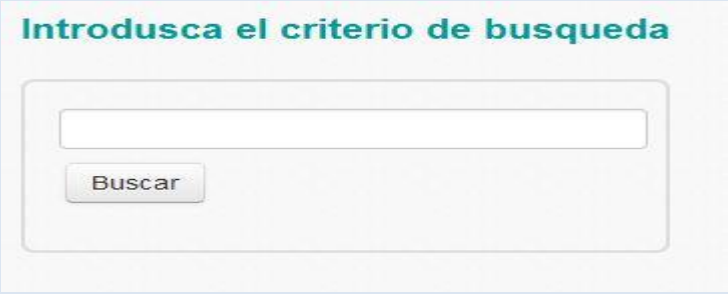
Programador responsable: Manuel Alejandro Padrón del Pico
Descripción: El usuario puede realizar búsquedas de las personas. Datos para la búsqueda: • Nombre (Obligatorio) • Usuario (Obligatorio)
Observaciones: El usuario debe introducir el nombre o usuario de la persona que desea buscar. En caso de que no exista se le mostrará un mensaje informándole que no existe ninguna persona con ese nombre.
Prototipo de interfaces: 

Tabla3:HU_1. Gestionar criterio de búsqueda.

Historia de Usuario	
Número: HU_2	Nombre: Visualizar persona
Usuario: Usuario público, Administrador	Iteración Asignada: Estimación2
Prioridad en Negocio: Alta	Riesgo de desarrollo: Alto
Puntos Estimados: 2	
Programador responsable: Manuel Alejandro Padrón del Pico	
Descripción El usuario público y administrador una vez insertado los datos a buscar podrá visualizarlos.	

Tabla 4:HU_2. Generar criterio.

Historia de Usuario	
Número: HU_3	Nombre: Autenticar usuario
Usuario: Administrador	Iteración Asignada: Estimación4
Prioridad en Negocio: Alta	Riesgo de desarrollo: Alto
Puntos Estimados: 1	
Programador responsable: Manuel Alejandro Padrón del Pico	
Descripción: El usuario administrador podrá autenticarse insertando los siguientes datos: • Usuario • Contraseña Luego que el usuario es autenticado podrá hacer los cambios necesarios de los datos, estos son verificados y se inicia una sesión.	
Observaciones: En caso de que los datos del usuario autenticado no coincidan con los registrados en la aplicación se mostrará un mensaje indicando que falló la autenticación.	
Prototipo de interfaces: 	

Tabla 5:HU_3. Autenticar usuario

Historia de Usuario	
Número: HU_4	Nombre: Gestionar usuario.
Usuario: Administrador	Iteración Asignada: Estimación 3
Prioridad en Negocio: Media	Riesgo de desarrollo: Medio

Puntos Estimados: 1
Programador responsable: Manuel Alejandro Padrón del Pico
Descripción: Esta funcionalidad describe como se llevará a cabo el proceso de gestionar el usuario que se registre en la aplicación, donde el administrador podrá adicionar, eliminar, mostrar un determinado usuario. Datos de usuarios necesarios para registrarse: • Usuario (Obligatorio) • Contraseña (Obligatorio)
Observaciones: En caso de que sea un usuario "registrado" solo podrá ver y editar su perfil. Si es administrador tendrá acceso a todas las opciones de gestionar usuario.
Prototipo de interfaces:

Tabla 6:HU_4. Gestionar usuario

La planificación se puede realizar basándose en el tiempo o el alcance. La velocidad del proyecto es utilizada para establecer cuántas HU se pueden implementar antes de una fecha determinada o cuánto tiempo tomará implementar un conjunto de HU. Al planificar por tiempo, se multiplica el número de iteraciones por la velocidad del proyecto, determinándose cuántos puntos se pueden completar. A continuación se muestran las estimaciones realizadas para la implementación de las HU **(Penadés)**.

No.	Historias de Usuario	Puntos Estimados
1	Gestionar Criterios de búsqueda	1
2	Autentificar usuario	1
3	Generar Criterio	2
4	Gestionar Usuario en BD	1

Tabla 7: Puntos Estimación

2.8 Plan de Iteración

La metodología XP aporta mayor valor a la colaboración con el cliente y al desarrollo incremental del software con iteraciones muy cortas. Al comienzo de cada ciclo, se realiza una reunión de planificación de la iteración. Cada HU se traduce en tareas específicas de programación. Así mismo, para cada HU se establecen las pruebas de aceptación. Estas pruebas se realizan al final del ciclo en el que se desarrollan, pero también al final de cada uno de los ciclos siguientes, para verificar que subsiguientes iteraciones no han afectado a las anteriores. Al planificar según alcance del sistema, se divide la suma de puntos de las HU seleccionadas entre la velocidad del proyecto, obteniendo el número de iteraciones necesarias para su implementación. Todo proyecto que siga la metodología XP se ha de dividir en iteraciones de aproximadamente tres semanas de duración. Al comienzo de cada iteración el cliente debe seleccionar las HU definidas en el plan de iteraciones que serán implementadas **(Beck, 1996)**.

Iteración 1: En la primera iteración se realiza la implementación de las HU Buscar datos y Visualizar datos con prioridad de negocio alta, obteniendo al final de la misma una primera versión de prueba y dando a la aplicación las primeras funcionalidades.

Iteración 2: En la segunda iteración se implementa la HU Visualizar datos con prioridad de negocio alta. Además se corregirán errores o disconformidades del cliente con las HU implementadas en la anterior iteración. De esta forma se obtiene la segunda versión de prueba de la aplicación.

Iteración 3: En la tercera iteración, ya implementadas las funcionalidades especificadas se realiza el desarrollo de las HU con prioridad media, se corrigen errores de esta e iteraciones anteriores. Al final de esta iteración se obtendrá la versión final de la aplicación.

2.8.1 Plan de duración de las iteraciones

Este plan se realiza con el objetivo de reflejar cuáles serán las HU que serán implementadas en cada una de las iteraciones, así como el tiempo destinado a cada una de ellas y el orden en que se implementarán.

Iteración	Orden de las Historias de Usuario	Duración de las iteraciones
Iteración 1	Gestionar Criterios de búsqueda	3 semanas
Iteración 2	Generar Criterio	2 semanas
Iteración 3	Autenticar usuario Gestionar usuario en BD	3 semanas
Total		8 semanas

Tabla 8: Plan de duración de las iteraciones

2.9 Plan de entrega

El objetivo de este plan es producir rápidamente versiones de la aplicación que sean operativas, aunque estas no cuenten con toda la funcionalidad pretendida, pero sí que constituyan un resultado de valor para el negocio.

El plan de entrega que se muestra a continuación propone el tiempo aproximado de entrega de versiones de la aplicación implementadas durante cada una de las iteraciones.

Módulo	Primera iteración 4ta semana de Febrero	Segunda iteración 2da semana de Abril	Tercera iteración 4ra semana de Mayo
Versión	0.1	0.2	0.1

Tabla 9: Plan de entrega

2.10 Fases de Diseño

El diseño es fundamental, a diferencia de otras metodologías se realiza durante todo el tiempo de vida del proyecto por lo que se establecen los mecanismos, para que este sea revisado y mejorado continuamente según se van añadiendo funcionalidades al mismo.

En la fase de diseño XP propone un nuevo concepto llamado Metáfora[2].Su principal objetivo es mejorar la comunicación entre todos los integrantes del equipo, al crear una visión global y común de lo que se quiere desarrollar. Otro de los elementos importantes en XP es lograr un diseño sencillo con el menor número de clases y métodos, pero que funcione para todas las pruebas.

2.11 Tarjetas CRC

La metodología XP requiere de muy poco la representación del sistema mediante diagramas de clases utilizando la notación UML. En su lugar se usan las tarjetas CRC (Clase, Responsabilidad y Colaboración). Las cuales ayudan al equipo a definir actividades durante el diseño del sistema. Cada tarjeta representa una clase en la programación orientada a objetos. Las tarjetas CRC se componen de los siguientes elementos:

Clase: es cualquier persona, cosa, evento, concepto, pantalla o reporte.

Responsabilidades: es lo que debe hacer la clase, sus atributos y métodos.

Colaboradores: son el resto de las clases con las que interactúa para llevar a cabo sus responsabilidades.

Tarjeta CRC	
Clase: Principal/indexSuccess.php	
Responsabilidades	**Colaboraciones**
Gestionar criterio	actions.class

Tabla 10: Tarjeta CRC 1

Tarjeta CRC	
Clase: Persona/actions.class	
Responsabilidades	**Colaboraciones**
Gestionar criterio	1. PersonasPeer 2. EstudiantePeer 3. TrabajadorPeer 4. ProfesorPeer

Tabla 11: Tarjeta CRC 2

[2] Una metáfora es una historia que todo el mundo puede contar acerca de cómo funciona el sistema **(Solís, Noviembre 20, 2012).**

Tarjeta CRC	
Clase: buscarSuccess.php	
Responsabilidades	**Colaboraciones**
Generar criterios	buscarSuccess

Tabla 12: Tarjeta CRC 3

Tarjeta CRC	
Clase: conection.php	
Responsabilidades	**Colaboraciones**
Gestionar usuario en BD	actions.class

Tabla 13: Tarjeta CRC 4

Tarjeta CRC	
Clase: formnuevapersona.php	
Responsabilidades	**Colaboraciones**
Adicionar nueva persona	indexsucces.class

Tabla 14: Tarjeta CRC 5

Tarjeta CRC	
Clase: formeliminarpersona.php	
Responsabilidades	**Colaboraciones**
Eliminar nueva persona	indexsucces.class

Tabla 15: Tarjeta CRC 6

Para mejor comprensión de las clases representadas anteriormente en las tarjetas CRC se muestra el siguiente diagrama con las funcionalidades de la aplicación y con las cuales se implementan las HU. Las relaciones representan las colaboraciones entre clases.

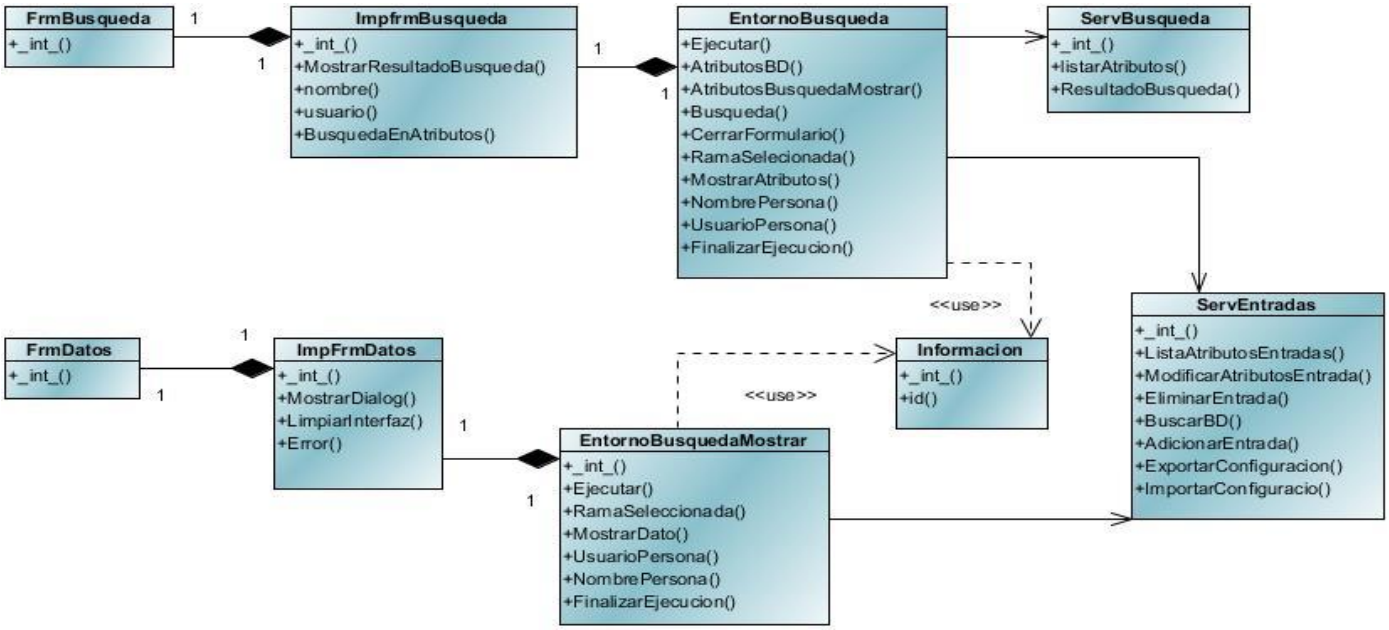


Figura 2: Diagrama de Clases

2.12 Estilo Arquitectónico Modelo- Vista- Controlador (MVC)

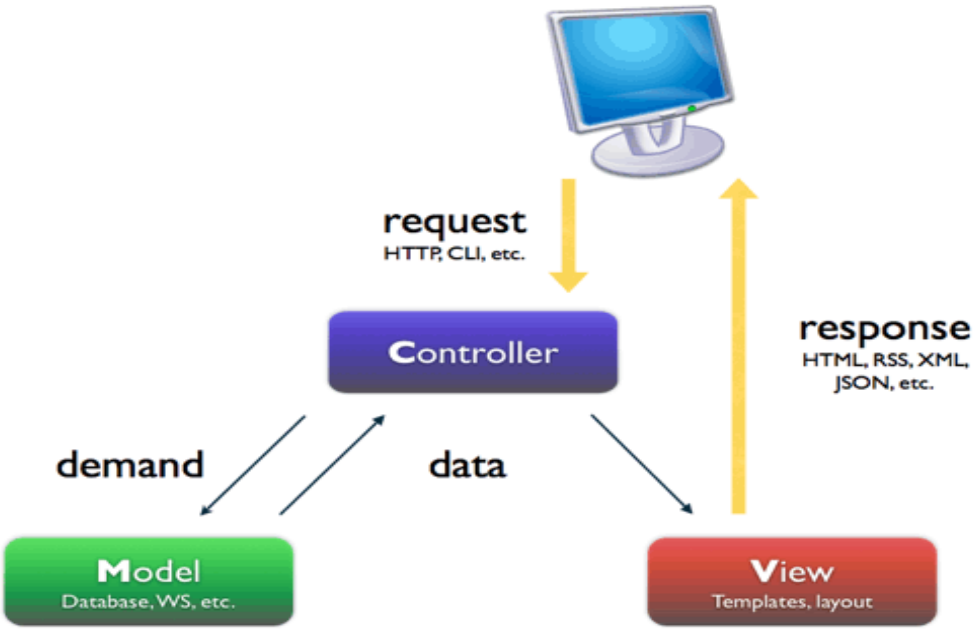


Figura 3: Modelo Vista Controlador

Modelo: El paquete "*model*" contiene las clases que permiten acceder y obtener información de los usuarios de la aplicación que se almacena en la base de datos.

Vista: Está compuesto por el paquete "*views*", el cual contiene páginas que controlan el desarrollo de la aplicación. Además en este paquete interactúan con las interfaces que despliegan la información al usuario, diseño que se logra a partir del uso de la librería y la aplicación de estilos CSS.

Controlador: Realiza llamadas al repositorio de datos para obtener la información y enviarlos a la vista para que los muestre al usuario.

2.13 Patrones de diseño

El uso de patrones para el desarrollo de aplicaciones informáticas, es una buena práctica, aportando seguridad y robustez en la misma. Además de facilitar el trabajo, son de gran importancia en todos los proyectos informáticos, garantizando claridad en la estructura de la aplicación y como consecuencia mayor organización. El sistema que se desea implementar basa su arquitectura empleando patrones

arquitectónico y de diseños, en la definición de las clases del sistema y el diseño del sistema a implementar, compuesta por una arquitectura en tres niveles o capaz. Para ello tiene suma importancia la utilización de los patrones GRASP **(Wesley, 2009)**.

2.13.1 Patrones GRASP (Patrones de Software para la Asignación General de Responsabilidad)

"Los patrones *GRASP* describen los principios fundamentales de diseño de objetos para la asignación de responsabilidades. Constituyen un apoyo para la enseñanza que ayuda a entender el diseño de objeto esencial y aplica el razonamiento para el diseño de una forma sistemática, racional y explicable." Estos patrones se describen a continuación **(Márquez, 2008)**.

Bajo acoplamiento:

- ✓ El Bajo Acoplamiento es un patrón evaluativo que el diseñador aplica al juzgar sus decisiones de diseño. Estimula asignar una responsabilidad de modo que su colocación no incremente el acoplamiento tanto que produzca los resultados negativos propios de un alto acoplamiento**. (Buscar referencia)**

Alta Cohesión:

- ✓ Asignar una responsabilidad de modo que la cohesión siga siendo alta. Representa una Clase con responsabilidades moderadas en un área funcional, colaborando con otras para concretar tareas. Diseño más claro y comprensible. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme **(Márquez, 2008)**.

Experto:

- ✓ Es un principio básico que suele utilizarse en el diseño orientado a objetos. Con él no se pretende designar una idea oscura ni extraña; expresa simplemente la "intuición" de que los objetos hacen cosas relacionadas con la información que poseen. Ofrece una analogía con el mundo real **(Márquez, 2008)**.

Creador:

- ✓ El patrón Creador guía la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El propósito fundamental de este patrón es encontrar un creador que se debe conectar con el objeto producido en cualquier evento, se debe buscar una clase de objeto que agregue, contenga y realice otras operaciones sobre este tipo de instancias **(Márquez, 2008)**.

Controlador:

- ✓ Es un evento generado por actores externos. Se asocian con operaciones del sistema, operaciones del sistema como respuestas a los eventos del sistema, tal como se relacionan los mensajes y los métodos **(Márquez, 2008)**. Este patrón se pone de manifiesto con la utilización de una clase que coordina o controla las actividades que son necesarias realizar con las demás clases, en este caso Multi Decisión PAAT es quien hace uso de este patrón. No realiza mucho trabajo por sí mismo.

En la descripción del diagrama de clases se especifica cómo se materializa el uso de estos patrones en el diseño realizado.

2.14 Modelo de Datos

Es un conjunto de conceptos que permiten describir, a distintos niveles de abstracción, la estructura de una base de datos, la cual se denomina esquema. Según el nivel de abstracción, el modelo de datos que permite su descripción será un modelo externo, global o interno, cada uno de los cuales ofrece distintos elementos de descripción. Los modelos externos permiten representar los datos que necesita cada usuario en particular con las estructuras propias del lenguaje de programación que va a emplear. Los modelos globales ayudan a describir los datos para el conjunto de usuarios, se podría decir que es la información a nivel de empresa y por último, los modelos internos que están orientados a la máquina, siendo sus elementos de descripción punteros, índices, agrupamientos **(Material, 2013)**.

Existen 3 tipos de modelos de datos:

- ✓ **Los modelos conceptuales** (también denominados de alto nivel) facilitan la descripción global del conjunto de información de la empresa con independencia de la máquina, por lo que sus conceptos son cercanos al mundo real.

- ✓ **Los modelos convencionales** se encuentran soportados por los Sistemas de Gestión de Bases de Datos (SGBD) y están orientados a describir los datos a nivel lógico para el SGBD, por lo que sus conceptos son propios de cada SGBD**.**

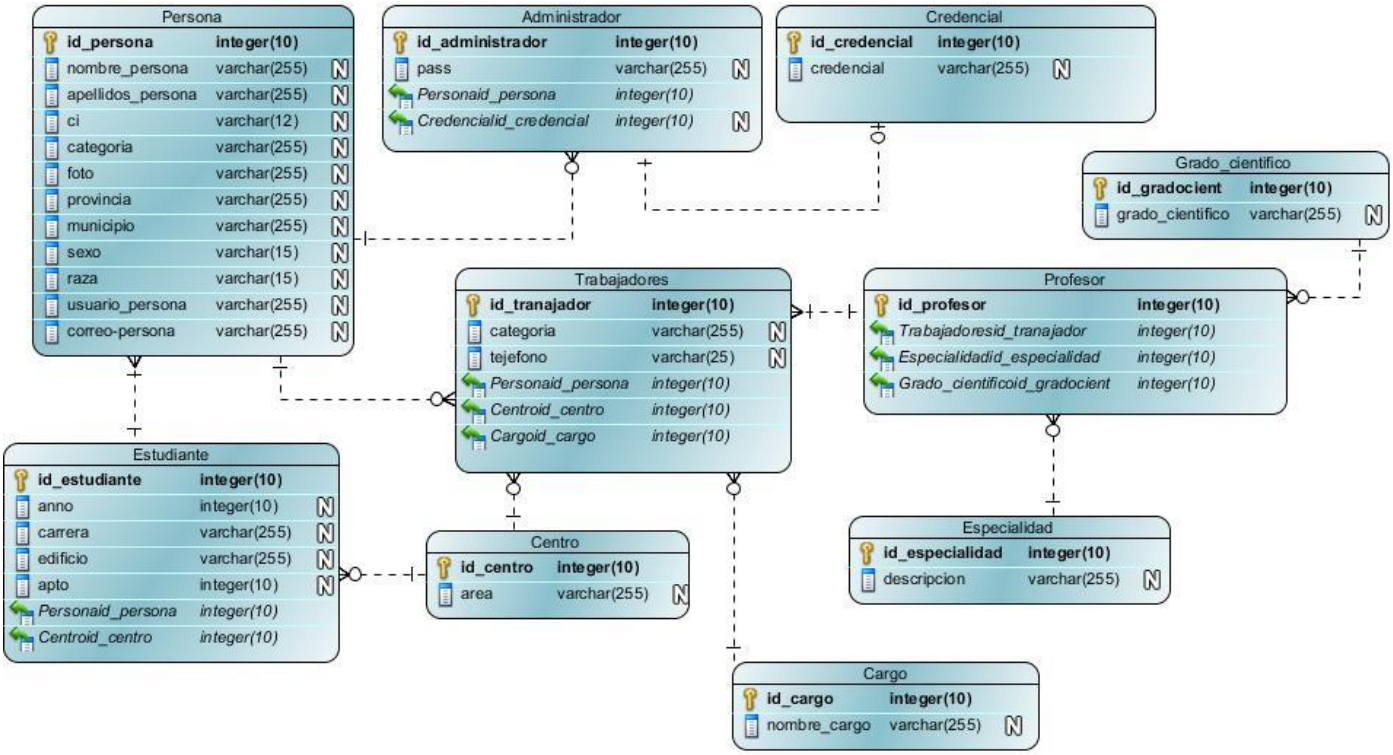


Figura 4: Modelo de Datos

Se describe el modelo de datos en el Anexo 2, cada una de las tablas y sus atributos, un ejemplo de la descripción de estos modelos es la tabla candidato.

Nombre Tabla: Estudiante		
Descripción: En esta tabla se almacenan los datos correspondientes a una decisión.		
Atributos	**Tipo**	**Descripción**
id_estudiante	Int	Es el identificador del estudiante. Es la llave primaria de la tabla.
anno	Int	Describe el anno del estudiante.
carrera	String	Describe la carrera que estudia el estudiante
Id_persona	Int	Es el identificador de la persona. Hace referencia de la clase padre.

Tabla 16: Tabla Decisión de la Base de Datos

Conclusiones Parciales

En el capítulo se realizó la propuesta de solución para implementar la aplicación:

- ✓ Se definieron los requisitos funcionales y no funcionales para la misma.
- ✓ Fueron elaborados y descritos los artefactos que propone la metodología XP para las fases Planificación y Diseño.
- ✓ Se confeccionaron 3 Historias de usuarios, de las cuales 1es de prioridad muy alta, se planificó su implementación durante 3 iteraciones comenzando por las HU de mayor prioridad atendiendo a su importancia en el negocio.
- ✓ Se elaboraron 6 tarjetas CRC y se ejemplificó la aplicación de los patrones de diseño, así como un diagrama con la estructura de la aplicación aplicando el patrón arquitectónico MVC.

Capítulo 3 Implementación y Prueba

La implementación en el proceso de desarrollo de un software adquiere gran importancia debido a que le da funcionalidad al producto que se desarrolla. Además, son importantes las pruebas que se le realizan al mismo para validar su correcto funcionamiento. El capítulo abarca las fases de implementación y prueba de la aplicación. Se exponen las tareas asignadas a las HU para llevar a cabo la implementación. Se muestra el estándar de codificación utilizado y el código de las HU con mayor prioridad de desarrollo. Se describen las pruebas realizadas a la aplicación con el objetivo de verificar si se cumplieron los requerimientos de la misma.

3.1 Implementación

3.3.1 Diagrama de componentes

Un Diagrama de Componente es, como su nombre lo indica, un esquema o diagrama que muestra las interacciones y relaciones de los componentes de un modelo. Entendiéndose como componente a una clase de uso específico, que puede ser implementada desde un entorno de desarrollo, ya sea de código binario, fuente o ejecutable; dichos componentes poseen tipo, que indican si pueden ser útiles en tiempo de compilación, enlace y ejecución. Este tipo de diagrama se representa mediante componentes unidos mediante relaciones de dependencia (generalmente de compilación) **(Web, 2010, En línea)**.

Los Diagramas de Componentes ilustran las piezas del software, controladores embebidos[3] que conformarán un sistema. Un diagrama de Componentes tiene un nivel más alto de abstracción que un diagrama de clase, usualmente un componente se implementa por una o más clases en tiempo de ejecución. Estos son bloques de construcción, como eventualmente un componente puede comprender una gran porción de un sistema.

[3] Un **sistema embebido o empotrado** es un sistema de computación diseñado para realizar una o algunas pocas funciones dedicadas frecuentemente en un sistema de computación en tiempo real **(Barr, 21-04-2007)**.

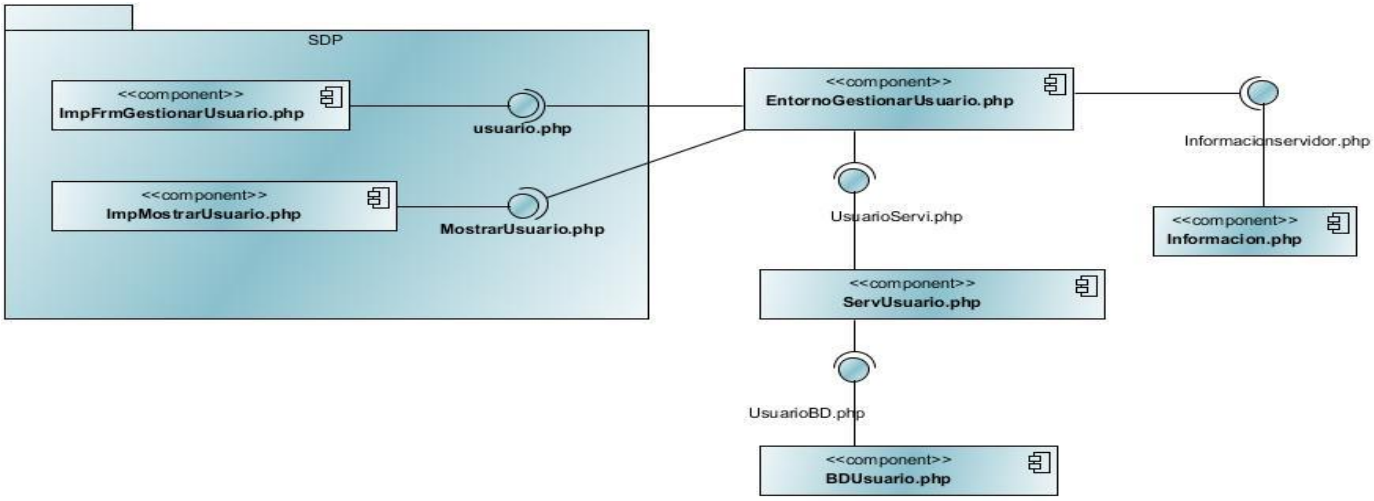


Figura 5: Diagrama de componentes – Búsqueda Avanzada

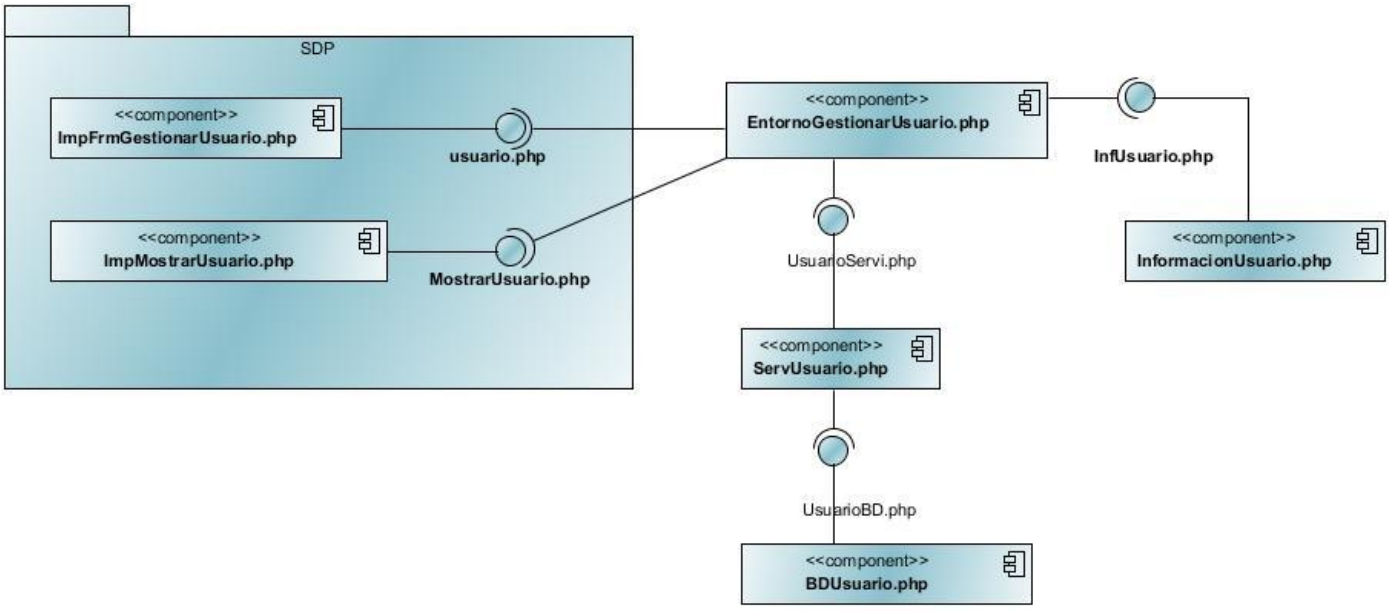


Figura 6: Diagrama de componentes – Gestionar Usuario

3.2 Desarrollo de las iteraciones

En la fase de Planificación se detallaron las HU correspondientes a cada una de las iteraciones a desarrollar, teniendo en cuenta las necesidades requeridas por el cliente. Estas HU se

descomponen en tareas de programación o ingeniería, que se asignan a un equipo de desarrollo o persona. Estas tareas no tienen que necesariamente ser entendidas por el cliente, pueden ser escritas en lenguaje técnico y son para el uso estricto de los programadores. A continuación se detallan cada una de las iteraciones y tareas de ingeniería asignadas a cada HU.

Iteración 1

En la siguiente iteración se le da cumplimiento a la implementación de las HU Gestionar Criterios de búsqueda.

1. **HU_1: Gestionar Criterio**

Tarea de ingeniería	
No. De la Tarea:T_1	**No. De la HU:**HU_1
Nombre de la Tarea: Diseñar la interface requerida para la funcionalidad de la gestión de criterios de búsqueda	
Tipo de Tarea: Desarrollo	**Puntos estimados:**0.1
Fecha inicio: 26-Febrero-2013	**Fecha fin:**29-Febrero-2013
Programador Responsable: Manuel Alejandro Padrón del Pico	
Descripción: Se debe crear una interfaz donde el usuario inserte un criterio de búsqueda (nombre o usuario) según la persona que desee encontrar	

Tabla 17: Tarea de Ingeniería 1

Tarea de ingeniería	
No. De la Tarea: T_2	**No. De la HU:** HU_1
Nombre de la Tarea: Obtener resultados de búsqueda	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 2
Fecha inicio: 11-Abril-2013	**Fecha fin:** 13-Abril-2013
Programador Responsable: Manuel Alejandro Padrón del Pico	
Descripción: Teniendo en cuenta el parámetro introducido por el usuario se realiza la búsqueda en el repositorio.	

Tabla 18: Tarea de Ingeniería 2

Iteración 2

En esta iteración se desarrolla la HU 2. Esta HU visualiza una búsqueda

2. HU_2: Generar Criterio

Tarea de ingeniería	
No. De la Tarea: T_3	**No. De la HU:** HU_2
Nombre de la Tarea: Visualizar resultados de búsqueda	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 2
Fecha inicio: 23-Abril-2013	**Fecha fin:** 29-Abril-2013
Programador Responsable: Manuel Alejandro Padrón del Pico	
Descripción: Luego de realizada la búsqueda y encontrar resultados se muestra una lista de nombre (es), introducido por el usuario.	

Tabla 19: Tarea de Ingeniería 3

Iteración 3

En esta última iteración se desarrollan las HU número 3 y 4 las cuales permiten la autenticación y gestión de los usuarios. Las tareas de ingeniería asociadas a las iteraciones 3 y 4 se encuentran descritas en el Anexo 3.

3.3 Estándares de codificación

XP propone la definición de un estándar de programación para mantener un código legible. Esto beneficia la comunicación de los programadores a través del código y aún más si la implementación se realiza en pareja. Las convenciones de código, o como también se conocen estándares de codificación son modelos de programación que no están enfocadas a la lógica del programa, sino a su estructura y apariencia física para facilitar la lectura, comprensión y mantenimiento del código. A continuación se presentan algunas de estas convenciones para la programación de la aplicación **(Solís, 2003)**:

Indentación[4]

La unidad de indentación de bloques de sentencias son 4 espacios.

Las demás estándares de codificación se encuentran descritas en el Anexo 4.

3.4 Pruebas de aceptación o caso de prueba

3.4.1 Diseño de los casos de prueba

Las pruebas de caja negra también conocidas con sus varios nombres como pruebas funcionales, pruebas de caja opaca, pruebas de entrada/salida, pruebas inducidas por los datos, son las que no toman en cuenta el código como quien dice el que lo prueba no sabe cómo está estructurado por dentro el programa o bien no necesita saber nada de programación, solo necesita saber cuáles pueden ser las posibles entradas sin necesidad de entender cómo se deben obtener las salidas, donde se trata de encontrar errores en la interfaz mientras se está usando, el cómo luce, se maneja **(Presman, 2003)**. Este grupo de pruebas se utilizó para a validación de las funcionalidades del sistema. Las pruebas de caja negra se centran principalmente en lo que *"se quiere"* de un módulo o sección específica de un software, es decir, es una manera de encontrar casos específicos en ese modulo que atiendan a su especificación.

Figura 7: Esquema caja negra

Según **(Presman, 2003)** las pruebas de caja negra permiten identificar problemas tales como:

- ✓ Funciones incorrectas o ausentes.
- ✓ Errores de interfaz.
- ✓ Errores en estructuras de datos o en accesos a las Bases de Datos externas.
- ✓ Errores de rendimiento.
- ✓ Errores de inicialización y terminación.

[4] **Indentación** o **identación** es un anglicismo (de la palabra inglesa *indentation*) de uso común en informática que significa mover un bloque de texto hacia la derecha insertando espacios o tabuladores para separarlo del texto adyacente, lo que en el ámbito de la imprenta se ha denominado siempre como *sangrado* o *sangría*.

3.5 Diseño de los casos de prueba

Las pruebas de aceptación son creadas sobre la base de las HU. El cliente debe especificar uno o diversos escenarios para comprobar que las HU han sido correctamente implementadas. Las mismas son consideradas como “pruebas de caja negra”. Los clientes son responsables de verificar que los resultados de estas pruebas sean correctos. En caso de que fallen varias pruebas, deben indicar el orden de prioridad de resolución. Una HU no se considera terminada hasta que pase todas las pruebas de aceptación. Para asegurarse que la aplicación desarrollada cumpla sus requisitos, a continuación se representan algunas de las pruebas de aceptación realizadas a las HU, el resto se encuentran en el expediente de proyecto en la planilla: Casos de pruebas de aceptación.

Caso de Prueba de Aceptación	
No. De la Tarea: A_2	**Nombre de Historio de Usuario:** Generar criterios
Nombre de la persona que realiza la Tarea: Manuel Padrón del Pico	
Descripción de la Prueba: Se ejecuta la prueba y se verifica que se ordene correctamente.	
Condiciones de Ejecución: • Que se halla cargado previamente toda información.	
Entrada / Pasos de ejecución: Para realizar esta historia es necesario que se halla persistido satisfactoriamente toda la información antes mencionada. Luego de haber verificado esto se le muestra al especialista una tabla con las alternativas ordenadas descendentemente.	
Resultado Esperado: Que se muestren los datos adecuadamente	
Evaluación de la Prueba: Satisfactorio	

Tabla 20: Diseño de Caso de Prueba de Aceptación 1

En el anexo 5 se encuentra el resto de las pruebas de aceptación realizadas.

Conclusiones Parciales.

- ✓ Se describieron las distintas tareas de ingeniería que ayudaron a implementar las HU.

- ✓ Se definió el estándar de codificación para lograr mejor comunicación entre los miembros del equipo y mayor legibilidad en la implementación.
- ✓ Se presentaron fragmentos de código de varias funcionalidades de la aplicación.
- ✓ Se realizaron las pruebas de aceptación para obtener un producto de mayor calidad. Donde se obtuvieron todas las pruebas satisfactorias.

Conclusiones Generales.

Una vez terminado el presente trabajo se puede concluir que fueron cumplidos los objetivos planteados. Para los cuales se arrojan las siguientes conclusiones generales:

- ✓ Se logró dar cumplimiento a las tares descritas al iniciarse la presente investigación.
- ✓ Mediante la utilización de un sistema búsqueda de personas se puede optimizar las tareas y ahorrar tiempo. Además de que puede ser usado en diversas ramas de la vida y en cualquier renglón de la sociedad.
- ✓ Se realizó una descripción de las herramientas a utilizar en la implementación de la propuesta, permitiendo a los desarrolladores una mejor comprensión de dichas herramientas y tecnologías.
- ✓ El sistema se desarrolló utilizando el IDE NetBeans para PHP para la implementación de cada una de las clases de los marcos de trabajo: Symfony para el negocio, Propel para el acceso a datos y XP como metodología de desarrollo de software. Como sistema gestor de base de datos PostgreSQL.
- ✓ Con la aplicación de la metodología se fueron generando los artefactos que la misma exigía, lo cual ayuda al entendimiento del sistema por parte del cliente.
- ✓ El sistema fue sometido a rigurosas pruebas de funcionalidad y de validaciones del diseño las cuales arrojaron que la aplicación cuenta con una excelente calidad. Además de dar cumplimiento a los requisitos del cliente definidos en la etapa inicial de la investigación.

Recomendaciones

Se recomienda:

- ✓ Realiza las modificaciones para su extensión en la UCLV.
- ✓ Realizar la gestión para integrarlo a los servicios LDAP, una vez arreglado los problemas de actualización de sus plantillas.
- ✓ Desarrollar una versión para la apelación desktop del Servicio de Directorio de Personas.
- ✓ Realizar una versión para telefonía móvil.

.

Referencias Bibliográficas

2011. Introducción a UML.
AUTORES, C. D. 2009. Los Servicio de Directorio, una ventana al futuro

BARR, M. 21-04-2007. Embedded Systems Glossary. Netrino Technical Library.
BECK, K. 1996. Programming eXtreme (XP).
BORGE, P. V. 2007. Implementacion de un servicio de directorio en la red.
CARTER, G. 2011. LDAP System Adnministration.
CONSORTIUM, W. W. W. 20-06-2009. What is CSS?
CROCKFORD, D. Citado 2012. Functional JavaScript.
DORADO, M. D. Novienbre, 2005. Todo Programación. NetBeans IDE. La alternativa a Eclipse No. 13, Páginas 32-34.
ECURED 28-02-2013. Artículo.
F., D. 1998. Objetos, componentes y Estructuras con UML, The Catalysis Aproach, .
HURTADO, B. G. 2009. El Proceso de análisis jerárquico (AHP) como herramienta para latoma de desiciones en la selección de proveedores.
HURTADO, G. 2005. El proceso de analisis jerarquico (AHP) como herramienta para la toma de desiciones en la seleccion de proveedores.
IBM Unified Modeling Language (UML).
IBM 2011. Unified Modeling Language (UML).
LÍNEA, E. 2011. *Introducción a UML* [Online]. Available: http://docs.kde.org/stable/es/kdesdk/umbrello/uml-basics.html#about-um.
MÁRQUEZ, I. D. 2008. Diseño e implementación de las capas de negocio y acceso a datos de los módulos Planificación y Ejecución de Vistas Familiares.
MATERIAL 2013. Material docente de libre consulta OpenCourseWare 6. Modelado de datos.
NUÑEZ, J. 30-07-2009. ¿Por que un framework?
PENADÉS, P. L. A. C. M. Metodologías agiles para el desarrollo de los software. eXtreme Programing II.
PICO, M. A. P. D. 2012. Investigación realizada a la decana y profesores de la facultad, mediante pregunta y respuestas.
PRESMAN, R. S. 2003. Ingeniería del Software, un enfoque Práctica. El Proceso. 5to. .
R., L. En línea. *Buenas Tareas. Métodos Teóricos* [Online]. Available: http://www.buenastareas.com/ensayos/Metodos-Teoricos/136411.html..
RAYMOND, E. Citado 2012. The Art of Unix Programming.
RODRÍGUEZ, E. 2012. *RE: Entrevista realizada al profesor.*
RODRIGUEZ, R. En linea. *Métodos empíricos* [Online]. Available: http://es.scribd.com/doc/21229743/METODOS-EMPIRICOS..
SOLÍS, C. M. Noviembre 20, 2012. Una explicación de la programación extrema XP.
SOLÍS, M. C. 2003. Una explicacion de la programación extrema (XP).
TEST, F. D. 13-01-2011. *PostgreSQL 9 el verdadero test* [Online].
ULLMAN, L. 2007. *PHP 5 advanced*.
WEB, S. 2010, En línea. *The AMPAC poeple* [Online]. Available: http://www.semichem.com/codessa/default.php.
WESLEY, A. 2009. Elements of Reusable Object-Oriented Software
XAMPP. Febrero, 2012. *XAMPP* [Online].

Bibliografías

- ✓ Sergio Luján Mora (2002) (en español, libro completo gratuito en pdf). *Programación de aplicaciones web: historia, principios básicos y clientes web* (1ª edición). Editorial Club Universitario. http://hdl.handle.net/10045/16995.
- ✓ Tutoriales de programación PHP con NetBeans.
- ✓ Ingeniería del software. Un enfoque práctico (sexta edición), R. S. Pressman. McGraw Hill Higher Education. Sitio en Inglés
- ✓ Modelado de Sistemas con UML por Popkin Software and Systems (1ª edición). Editorial Club Universitaria CUJAE.
- ✓ Descripción de todas las especificaciones del CSS, por la W3C. Libro en Ingles.
- ✓ Flanagan, David (2002). *JavaScript: The Definitive Guide* (4 ª Edición edición).
- ✓ Sergio Luján Mora (2001) (en español, libro completo gratuito en pdf). *Programación en Internet: Clientes Web* (1ª edición). Editorial Club Universitario.
- ✓ Sergio Luján Mora (2002) (en español, libro completo gratuito en pdf). *Programación de aplicaciones web: historia, principios básicos y clientes web* (1ª edición). Editorial Club Universitario.
- ✓ El Tutorial Jobeet. Fabien Potencier.

Anexos

Anexo 1 Plantilla de Concepción Inicial del Sistema

1. *Polo productivo:*

Macro proyecto de investigación: Sistema para la aplicación de un servicio de directorio de personas.

2. *Clasificación del proyecto:*

Desarrollo de aplicación

3. *Tipo de proyecto:*

Centralizada en la facultad de MFC.

4. *Resumen:*

En el documento se refleja la visión general del producto a implementar, además de realizar la descripción de los roles que intervienen en el desarrollo del software, así como las responsabilidades que tendrán en dicho proceso. Se documenta el tipo de proyecto al que pertenece. Se recoge además, cuales herramientas serán utilizadas para el desarrollo de la aplicación, el alcance que va a tener, una descripción de los involucrados en el negocio, cuales son los motivos de la necesidad del desarrollo del software y la propuesta de solución.

5. *Surgimiento.*

La necesidad de realizar el sistema surge luego de haber realizado un análisis de la falta de información que existe del personal que radica en las distintas áreas de la Universidad. Esto traía como consecuencia que el proceso lejos de brindar criterios eficientes y confiables, se volvía dudoso, lo que provocaba que en muchas ocasiones no se realizara una correcta comunicación.

6. *¿Qué es?*

El sistema es una herramienta de apoyo a los servicios que brinda el servidor LDAP, el cual ofrece criterios de selección en ambientes donde la realización del mismo de forma manual se torna impracticable y susceptible a errores.

7. Metodología a utilizar:

XP es la metodología que consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo al usuario final, pues es uno de los requisitos para llegar al éxito del proyecto y está basada completamente en los valores y principios de las metodologías ágiles. Por lo que XP respalda con sus prácticas todo el ciclo de desarrollo y de esta forma se obtiene un proceso de software completo.

8. Roles.

Rol	Responsabilidad	Nombre
Líder del Proyecto	Asegurar que el proyecto se está llevando a cabo de acuerdo con las prácticas y que todo funcione según lo planeado. Su principal trabajo es remover los impedimentos y definir, así como reducir los riesgos del producto. Asegurar que se consigue los objetivos de la reunión de planificación de la iteración. Determina cuándo es necesario realizar algún cambio para lograr los objetivos de cada iteración.	Ernesto Rodríguez Rodríguez
Cliente	El cliente contribuye a definir las historias de usuario y los casos de prueba de aceptación, para validar su implementación. Además, asigna la prioridad a las historias de usuario y decide cuáles se implementan en cada iteración centrándose en aportar mayor valor al negocio y participa en la concepción inicial del sistema.	Servicio Gestor de Bases de Datos

Programadores	Es el encargado de producir el código y escribir las pruebas unitarias. Debe existir una comunicación y coordinación adecuada entre los programadores y otros miembros del equipo.	Manuel Alejandro Padrón Del Pico
Analista	Es el encargado de escribir las historias de usuario y las pruebas funcionales para validar su implementación. Además, asigna la prioridad a las historias de usuario y decide cuáles se implementan en cada iteración centrándose en aportar mayor valor al negocio, todo esto lo realiza junto con el cliente.	Manuel Alejandro Padrón Del Pico
Diseñadores	Es el encargado del diseño del sistema; así como el de los prototipos de interfaces, máximos responsables de la realización del diseño de las metáforas y supervisan el proceso de construcción.	Manuel Alejandro Padrón Del Pico
Encargado de Pruebas	Es el encargado de ayudar al cliente a escribir las pruebas funcionales. Ejecuta las pruebas regularmente, difunde los resultados en el equipo y es responsable de las herramientas de soporte para pruebas.	Manuel Alejandro Padrón Del Pico
Arquitecto	Se vincula directamente con el analista y el diseñador debido a que su trabajo tiene que ver con la	Manuel Alejandro Padrón Del Pico

	estructura y el diseño en grande del sistema. Ayuda en el diseño de las metáforas.	
Consultor	Es un miembro externo del equipo con un conocimiento específico en algún tema necesario para el proyecto, además aportan ideas y experiencias para el beneficio del sistema en desarrollo.	Freddy Mederos Ernesto Rodríguez

Tabla 21: Tabla de Roles

9. Misión.

Brindar servicios de información de datos personales donde la confección manual del mismo se torna impracticable y es susceptible a errores.

10. Alcance

El desarrollo de las funcionalidades necesarias para hacer el análisis de la información y dar los criterios para la información de datos.

11. Herramientas utilizadas.

11.1 Herramienta Case

Visual Paradigm que es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. También proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML. Presenta licencia gratuita y comercial. Es fácil de instalar y actualizar y compatible entre ediciones. Generación de bases de datos; Transformación de diagramas de Entidad-Relación en tablas de base de datos; Importación y exportación de ficheros XML.

12. Lenguaje utilizado:

PHP es un lenguaje de programación de uso general de script del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico. Fue uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en el documento HTML en lugar de llamar a un archivo externo que procese los datos. El código es interpretado por un servidor web con un módulo de procesador de PHP que genera la página Web resultante. PHP ha evolucionado por lo que ahora incluye también una interfaz de línea de comandos que puede ser usada en aplicaciones gráficas independientes. PHP puede ser usado en la mayoría de los servidores web al igual que en casi todos los sistemas operativos y plataformas sin ningún costo. Fue creado originalmente por Rasmus Lerdorf en 1995. Actualmente el lenguaje sigue siendo desarrollado con nuevas funciones por el grupo PHP.

13. Marcos de trabajo

Symfony es un completo framework diseñado para optimizar el desarrollo de las aplicaciones web mediante algunas de sus principales características. Para empezar, 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. El resultado de todas estas ventajas es que no se debe reinventar la rueda cada vez que se crea una nueva aplicación web. Symfony está desarrollado completamente con PHP 5. Ha sido probado en numerosos proyectos reales y se utiliza en sitios web de comercio electrónico de primer nivel. Symfony es compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSQL, Oracle y Microsoft SQL Server. Se puede ejecutar tanto en plataformas *nix (Unix, Linux, etc.) como en plataformas Windows.

PostgreSQL es un sistema de gestión de base de datos relacional orientada a objetos y libre, publicado bajo la licencia BSD. Es dirigido por una comunidad de desarrolladores que trabajan de forma desinteresada, altruista, libre y/o apoyada por organizaciones comerciales. PostgreSQL permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos. Esta estrategia es superior al uso de bloqueos por tabla o por filas común en otras bases de datos. Implementa el uso de rollback's, subconsultas y transacciones, haciendo su funcionamiento mucho más eficaz, y ofreciendo soluciones en campos en las que MySQL no podría. Tiene la capacidad de comprobar la integridad referencial, así como también la de almacenar procedimientos en la propia base de datos, equiparándolo con los gestores de bases de datos de alto nivel, como puede ser Oracle. Soporta

transacciones, claves ajenas (con comprobaciones de integridad referencial). Tiene mejor soporte para triggers y procedimientos en el servidor.

14. Entornos de Desarrollo Integrado (IDEs)

NetBeans permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de software llamados *módulos*. Un módulo es un archivo Java que contiene clases de java escritas para interactuar con las APIs de NetBeans y un archivo especial (manifest file) que lo identifica como módulo. Las aplicaciones construidas a partir de módulos pueden ser extendidas agregándole nuevos módulos. Debido a que los módulos pueden ser desarrollados independientemente, las aplicaciones basadas en la plataforma NetBeans pueden ser extendidas fácilmente por otros desarrolladores de software. NetBeans es un proyecto de código abierto de gran éxito con una gran base de usuarios, una comunidad en constante crecimiento, y con cerca de 100 socios en todo el mundo. Sun MicroSystems fundó el proyecto de código abierto NetBeans en junio de 2000 y continúa siendo el patrocinador principal de los proyectos.

15. Solución propuesta.

Partiendo de que la información que se tiene del estudiante se encuentra digital en un grupo de sistemas desarrollados en la universidad y teniendo en cuenta los elementos antes expuestos, se hace necesario el desarrollo de una aplicación informática que implemente los algoritmos necesarios para la implementación y mejore los tiempos de ejecución del modelo.

Anexo 2 Descripción del Modelo de Datos

Nombre Tabla: Persona		
Descripción: En esta tabla se almacenan los datos correspondientes de las personas.		
Atributos	**Tipo**	**Descripción**
id_persona	Int	Es el identificador de las personas. Es la llave primaria de la tabla.
nombre_persona	String	Es el nombre de la persona.
apellidos_persona	String	Es el apellido la persona.
foto	String	Hace referencia de la persona.
usuario_persona	String	Campo que identifica el usuario de la persona.
ci	Int	El número del carnet de identidad
provincia	String	Referencia donde pertenece.
municipio	String	Referencia donde pertenece
categoría	String	Referencia de si es estudiante o trabajador

Nombre Tabla: Trabajador		
Descripción: En esta tabla se almacenan los datos correspondientes a una decisión.		
Atributos	**Tipo**	**Descripción**
id_trabajador	String	Es el identificador de la tabla. Es la llave primaria.
cargo_ocupa	String	Describe si tiene algún cargo en el centro.
especialidad	String	Describe si es profesor. ¿De qué es?
facultad	String	Referencia de que facultad pertenece.
lugar_trabajo	String	Describe donde trabaja.
id_persona	Int	Es el identificador de la persona. Hace referencia de la clase padre.

Nombre Tabla: Profesor		
Descripción: En esta tabla se almacenan los datos correspondientes a una alternativa		
Atributos	**Tipo**	**Descripción**
id_profesor	Int	Es el identificador de la alternativa. Es la llave primaria de la tabla.
grado_cientifico	String	Describe si es máster, doctor u otro.
id_persona	Int	Es el identificador de la persona. Hace referencia de la clase padre.

Tabla 22: Tablas de la Base de Datos

Anexo 3 Tareas de Ingenierías

Tarea de ingeniería	
No. De la Tarea: T_1	**No. De la HU:** Todas
Nombre de la Tarea: Generación de la base de datos	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 13-Enero-2013	**Fecha fin:** 20-Enero-2013
Programador Responsable: Manuel Alejandro Padrón del Pico	
Descripción: Se genera la base de datos a partir del diseño realizado, esta debe implementarse sobre el sistema gestor definido para el desarrollo de la aplicación.	

Tabla 23: Tarea de Ingeniería 1

Tarea de ingeniería	
No. De la Tarea: T_2	**No. De la HU:** Todas
Nombre de la Tarea: Montaje del ambiente de desarrollo con la integración de los marcos de trabajo seleccionados para el desarrollo	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 13-Enero-2013	**Fecha fin:** 20-Enero-2013
Programador Responsable: Manuel Alejandro Padrón del Pico	
Descripción: Instalar y configurar los frameworks definidos para el desarrollo de manera que se pueda comenzar la implementación de la aplicación.	

Tabla 24: Tarea de Ingeniería 2

Tarea de ingeniería	
No. De la Tarea: T_3	**No. De la HU:** Todas
Nombre de la Tarea: Realizar el mapeo de la base de datos en el framework Symfony.	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 13-Enero-2013	**Fecha fin:** 20-Enero-2013
Programador Responsable: Manuel Alejandro Padrón del Pico	
Descripción: Realizar el mapeo de la base de datos en el framework de acceso a datos seleccionado.	

Tabla 25: Tarea de Ingeniería 3

Tarea de ingeniería	
No. De la Tarea: T_4	**Número de la Historio de Usuario:** HU_1
Nombre de la Tarea: Diseñar las interfaces requeridas para la funcionalidad: gestionar criterio.	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 21-Febrero-2013	**Fecha fin:** 22-Febrero-2013
Programador Responsable: Manuel Alejandro Padrón del Pico	
Descripción: Crear las interfaces necesarias, teniendo en cuenta los prototipos definidos	

en la Historia de Usuario.

Tabla 26: Tarea de Ingeniería 4

Tarea de ingeniería	
No. De la Tarea: T_5	**Número de la Historio de Usuario:** HU_1
Nombre de la Tarea: Implementar funcionalidad de autenticación de usuarios	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 21-Febrero-2013	**Fecha fin:** 22-Febrero-2013
Programador Responsable: Manuel Alejandro Padrón del Pico	
Descripción: Crear las clases y algoritmos necesarios para la implementación de la funcionalidad de autenticación de usuario.	

Tabla 27: Tarea de Ingeniería 5

Tarea de ingeniería	
No. De la Tarea: T_6	**Número de la Historio de Usuario:** HU_2, HU_3, HU_4
Nombre de la Tarea: Diseñar las interfaces necesarias	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 22-Febrero-2013	**Fecha fin:** 22-Febrero-2013
Programador Responsable: Manuel Alejandro Padrón del Pico	
Descripción: Se diseñan las interfaces necesarias para la implementación de las funcionalidades, teniendo en cuenta los prototipos diseñados en las HU.	

Tabla 28: Tarea de Ingeniería 6

Tarea de ingeniería	
No. De la Tarea: T_7	**Número de la Historio de Usuario:** HU_1
Nombre de la Tarea: Implementar. Gestionar criterio.	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 2
Fecha inicio: 23-Febrero-2013	**Fecha fin:** 24-Febrero-2013
Programador Responsable: Manuel Alejandro Padrón del Pico	
Descripción: Se implementa la funcionalidad vinculada con la historia de usuario.	

Tabla 29: Tarea de Ingeniería 7

Tarea de ingeniería	
No. De la Tarea: T_8	**Número de la Historio de Usuario:** HU_2
Nombre de la Tarea: Implementar. Generar criterio.	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 2
Fecha inicio: 9-Abril-2013	**Fecha fin:** 11-Abril-2013
Programador Responsable: Manuel Alejandro Padrón del Pico	
Descripción: Se implementa la funcionalidad vinculada con la historia de usuario.	

Tabla 30: Tarea de Ingeniería 8

Tarea de ingeniería	
No. De la Tarea: T_9	**Número de la Historio de Usuario:** HU_3
Nombre de la Tarea: Implementar. Autentificar usuario.	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 21-Abril-2013	**Fecha fin:** 23-Abril-2013
Programador Responsable: Manuel Alejandro Padrón del Pico	

Descripción: Se implementa la funcionalidad vinculada con la historia de usuario.

Tabla 31: Tarea de Ingeniería 9

Tarea de ingeniería	
No. De la Tarea: T_10	**Número de la Historio de Usuario:** HU_4
Nombre de la Tarea: Implementar. Gestionar usuario.	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 24-Abril-2013	**Fecha fin:** 28-Abril-2013
Programador Responsable: Manuel Alejandro Padrón del Pico	
Descripción: Se implementa la funcionalidad vinculada con la historia de usuario.	

Tabla 32: Tarea de Ingeniería10

Anexo 4 Estándares de codificación

Declaraciones

Se realizará una declaración por línea.

Ejemplo:

```
/**
 * Get the [usuario_persona] column value.
 *
 * @return     string
 */
public function getUsuarioPersona()
{
    return $this->usuario_persona;
}
```

Figura 8: Declaración por línea

Inicialización

Inicializar las variables locales donde se declaran. La única razón para no inicializar una variable donde se declara es si el valor inicial depende de algunos cálculos que deben ocurrir.

Colocación

Poner las declaraciones solo al principio de los bloques (un bloque es cualquier código encerrado por llaves "{" y "}".) No esperar al primer uso para declararlas. Con la excepción de los ciclos.

Declaraciones de clases

Ningún espacio en blanco entre el nombre de un método y el paréntesis "(" que abre su lista de parámetros. La llave de apertura "{" aparece al final de la misma línea de la sentencia declaración. La llave de cierre "}" empieza una nueva línea indentada para ajustarse a su sentencia de apertura correspondiente, excepto cuando no existen sentencias entre ambas, que debe aparecer inmediatamente después de la de apertura "{".

Métodos

Los métodos se separan con una línea en blanco.

Sentencias simples

Cada línea debe contener como máximo una sentencia.

Ejemplo:

```
public function setIdEstudiante($v)
{
    if ($v !== null) {
        $v = (int) $v;
    }

    if ($this->id_estudiante !== $v) {
        $this->id_estudiante = $v;
        $this->modifiedColumns[] = EstudiantePeer::ID_ESTUDIANTE;
    }

    return $this;
} // setIdEstudiante()
```

Figura 9: Sentencia SetEstudiante

Sentencias compuestas

- ✓ Las sentencias compuestas son sentencias que contienen listas de sentencias encerradas entre llaves. Las sentencias encerradas deben indentar se un nivel más que la sentencia compuesta.

- ✓ La llave de apertura se debe poner al final de la línea que comienza la sentencia compuesta; la llave de cierre debe empezar una nueva línea y ser indentada al mismo nivel que el principio de la sentencia compuesta.

Ejemplo:

```
public function executeBuscar(sfWebRequest $request) {
    $arreglo = array();
    $trabajador = new Trabajador();
    $i = 0;
    $est = 0;

    $resultado = PersonaPeer::BuscarPersonaUsuario($request->getParameter("criterio"));
    foreach ($resultado as $valor) {
        $i++;
    }
    if ($i == 0) {
        $resultado = PersonaPeer::BuscarPersonaNombre($request->getParameter("criterio"));
        foreach ($resultado as $valor) {
            $i++;
        }
```

Figura 10: Sentencia Compuestas

La clase de sentencias **if-else** debe tener la siguiente forma:

```
    if ($trabajador->getCategoria() == 'Profesor') {

        $profesor = ProfesorPeer::BuscarProfesorTrabajador($trabajador->getIdPersona());
        $arreglo["" . $per->getIdPersona()] = array(
            "id" => $per->getIdPersona(),
            "nombre" => $per->getNombrePersona(),
            "categoria" => $per->getCategoria(),
            "especialidad" => $per->getEspecialidad(),
            "cargoOcupa" => $per->getCargoOcupa(),
            "apellido" => $per->getApellidoPersona(),
            "foto" => $per->getFoto(),
            "ci" => $per->getCi(),
            "usuario" => $per->getUsuarioPersona(),
            "provincia" => $per->getProvincia(),
            "municipio" => $per->getMunicipio(),
            "sexo" => $per->getSexo(),
            "categoriaTrabajador" => $trabajador->getCategoria(),
            "lugarTrabajo" => $trabajador->getLugarTrabajo(),
            "telefono" => $trabajador->getTelefono(),
            "gradoCientifico" => $profesor->getGradoCientifico()
        );
        $_SESSION['noFound']=1;
    }

else {
    $arreglo["" . $per->getIdPersona()] = array(
        "id" => $per->getIdPersona(),
        "nombre" => $per->getNombrePersona(),
        "categoria" => $per->getCategaria(),
        "especialidad" => $per->getEspecialidad(),
        "cargoOcupa" => $per->getCargoOcupa(),
```

Figura 11: Sentencia if – else

Una sentencia **for** debe tener la siguiente forma:

```
foreach ($resultado as $valor) {
    $i++;
}
if ($i == 0) {
    $resultado = PersonaPeer::BuscarPersonaNombre($request->getParameter("criterio"));
    foreach ($resultado as $valor) {
        $i++;
    }
```

Figura 12: Sentencia for

Líneas en blanco

Se debe usar siempre una línea en blanco en las siguientes circunstancias:

- ✓ Entre métodos.
- ✓ Entre las variables locales de un método y su primera sentencia.
- ✓ Antes de un comentario de bloque o de un comentario de una línea.

Métodos

Los métodos cuando son compuestos tendrán la primera letra en minúscula, y la primera letra de las siguientes palabras que lo forma en mayúscula.

Ejemplo:

```
/**
 * Get the [usuario_persona] column value.
 *
 * @return     string
 */
public function getUsuarioPersona()
{
    return $this->usuario_persona;
}
```

Figura 13: Método

Variables

Los nombres de las variables empezarán con minúscula en caso de que sean palabras compuestas deben ser sin espacios y con la primera letra de la segunda palabra en mayúscula. Los nombres de variables de un solo carácter solo se utilizarán para variables índices temporales.

Ejemplo:

```
$arreglo = array();
$trabajador = new Trabajador();
$i = 0;
$est = 0;
```

Figura 14: Variables

Anexo 5 Caso de Pruebas de Aceptación

Caso de Prueba de Aceptación	
No. De la Tarea: A_1	**Nombre de Historio de Usuario:** Gestionar criterios de búsqueda
Nombre de la persona que realiza la Tarea: Manuel Padrón del Pico	
Descripción de la Prueba: Se ejecuta la prueba y se verifica que el usuario experto introduzca los criterios correctamente así como asignarle el peso correspondiente a la decisión.	
Condiciones de Ejecución:	
Entrada / Pasos de ejecución: Cuando el sistema cargue la pantalla Gestionar Criterios el especialista le proporcionara al sistema los criterios y el peso asociado a la decisión.	
Resultado Esperado: Que el sistema persista correctamente los datos en la base de datos	
Evaluación de la Prueba: Satisfactorio	

Tabla 33: Caso de prueba de Aceptación 1

Caso de Prueba de Aceptación	
No. De la Tarea: A_2	**Nombre de Historio de Usuario:** Generar criterios
Nombre de la persona que realiza la Tarea: Manuel Padrón del Pico	
Descripción de la Prueba: Se ejecuta la prueba y se verifica que se ordene correctamente.	
Condiciones de Ejecución: • Que se halla cargado previamente toda información.	

Entrada / Pasos de ejecución: Para realizar esta historia es necesario que se haya persistido satisfactoriamente toda la información antes mencionada. Luego de haber verificado esto se le muestra al especialista una tabla con las alternativas ordenadas descendentemente.
Resultado Esperado: Que se muestren los datos adecuadamente
Evaluación de la Prueba: Satisfactorio

Tabla 34: Caso de prueba de Aceptación 2

Caso de Prueba de Aceptación	
No. De la Tarea: A_3	**Nombre de Historio de Usuario:** Gestionar usuarios
Nombre de la persona que realiza la Tarea: Manuel Padrón del Pico	
Descripción de la Prueba: Se ejecuta la prueba y se verifica que se modifiquen y se eliminen los usuarios.	
Condiciones de Ejecución: • Que hayan usuarios registrados en el sistema.	
Entrada / Pasos de ejecución: El usuario especialista o administrador podrá administrar los usuarios que podrán interactuar con la aplicación.	
Resultado Esperado: Que se gestionen correctamente los usuarios	
Evaluación de la Prueba: Satisfactorio	

Tabla 35: Caso de prueba de Aceptación 3

Anexo 6 Interfaces de Usuario

Figura 15. Introducir criterio de búsqueda.

Figura 16: Autentificación del administrador.

Figura 17: Agregar persona

Figura 18: Eliminar persona

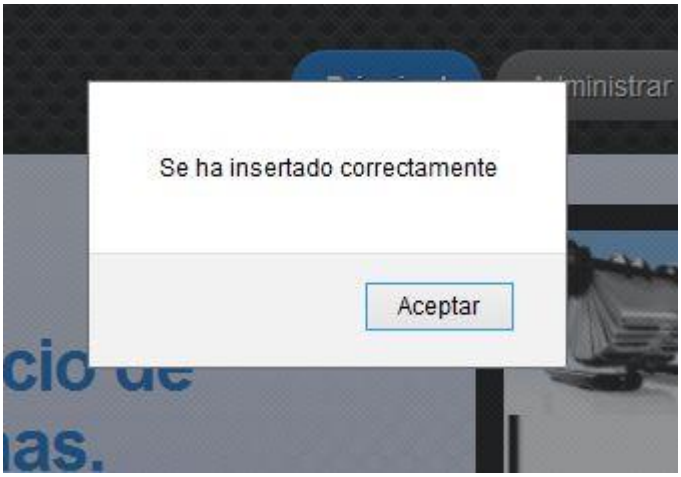


Figura 19: Cartel de alerta.