Universidad de las Ciencias Informáticas

Facultad 6

Sistema para la gestión de información académica en el Departamento Secretaría Docente de la ULEB Centro de Preparación de Operarios "Oscar Lucero Moya"

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autor: Félix Daniel Aguilera Pérez

Tutor: MSc. Lic. Ulises Llorente Pérez

Cotutora: Ing. Susej Beovides Luis

La Habana. Julio, 2015

Declaración de autoría

Declaro que soy autor único de este trabajo y autorizo a la UEB Centro de Preparación de Operarios "Oscar Lucero Moya" para que haga uso pertinente del mismo.

Para que así conste firmamos la presente a los___días del mes de_____del 2015.

Autor:

_____

Félix Daniel Aguilera Pérez

Tutor:

_____

MSc. Lic. Ulises Llorente Pérez

Cotutora:

_____

Ing. Susej Beovides Luis

Pensamientos

"Milito en el bando de los impacientes, milito en el bando de los apurados, de los que siempre presionan para que las cosas se hagan y de los que muchas veces tratan de hacer más de lo que se puede".

Fidel Castro

Agradecimientos

En especial a mi madre Bárbara Pérez Hernández porque sin su amor y guía no hubiese podido ser el hombre que soy hoy. Gracias por todo mami.

A mi padrastro Lázaro Montero Ledo por su apoyo durante toda mi carrera y su gran ayuda, a él le debo mucho.

A mi novia por su ayuda, dedicación y por esas largas horas que estuvo a mi lado sin dormir, fue como mi compañera de tesis.

A mi tutor y cotutora, por su ayuda paciente y por servirme de guía en el transcurso de este trabajo.

A mis profesores que contribuyeron en mi preparación para la vida.

A mis compañeros de aula, amigas y amigos quienes hemos estado juntos en las buenas y en las malas.

A mis compañeros de trabajo por toda su ayuda y en especial a: Anairis la tuti.

A las chicas del apartamento 112-204, un besito para ellas.

Y a modo general, a todas aquellas personas que me apoyaron de una manera u otra. Muchas gracias.

Dedicatorias

A mi madre Bárbara Pérez Hernández por ser la persona que más quiero en el universo, a la cual le debo todo lo que soy y lo que seré en esta vida. Tú, eres la persona que me motivó, ayudó y siempre me apoyó para cumplir este otro objetivo: hacerme Ingeniero en Ciencias Informáticas. Este trabajo de diploma es para ti mamá para que estés más orgullosa de tu hijo, te lo dedico de todo corazón. Te quiero mucho.

A la memoria de mi padre Félix Antonio Aguilera Vázquez que aunque ya no esté conmigo físicamente, siempre estará presente en todos los actos de mi vida. Te extraño mucho.

Resumen

El trabajo de diploma contiene la investigación para dar solución a la problemática de cómo contribuir a la mejora de la gestión de información académica en el Departamento Secretaría Docente de la UEB Centro de Preparación de Operarios “Oscar Lucero Moya”, lo cual permitió desarrollar una solución informática que contribuya a la mejora de la gestión de información académica en el Departamento Secretaría Docente del centro antes mencionado. Durante la investigación se realizó un estudio del estado del arte referente a los sistemas de gestión de información vinculados al campo de acción, de metodologías de desarrollo de software, herramientas y tecnologías a utilizar para la elaboración de la solución informática seleccionándose las más convenientes. La metodología de desarrollo de software Programación Extrema (XP) fue la guía para el desarrollo del sistema, generando los artefactos propuestos en cada iteración. Las pruebas realizadas al sistema permitió la obtención de un sistema libre de errores y la satisfacción del cliente.

Palabras claves: Centro de Preparación de Operarios, “Oscar Lucero Moya”, Secretaría Docente, UEB.

Índice

Introducción

Las organizaciones, en la era de la información, requieren agilidad, movilidad, innovación y cambios necesarios para enfrentar las nuevas amenazas y oportunidades en ambientes de intensa transformación y turbulencia. Los productos y servicios se adaptan de manera continua a las exigencias y necesidades de los clientes. Las personas, junto con sus conocimientos y habilidades intelectuales, se convierten en la base principal de la nueva organización.

La gestión de la información se ha convertido en un proceso difícil, pues la capacidad para generar y almacenar este recurso empresarial ha crecido significativamente a nivel mundial, lo que dificulta su análisis mediante los métodos tradicionales existentes, de ahí el surgimiento de nuevas técnicas y herramientas como son los sistemas de gestión de información.

El desarrollo del software en Cuba se ha visto incrementado en los últimos años como parte del auge a nivel mundial, de ahí el surgimiento de centros de desarrollo de software a lo largo de todo el país, que vinculan la teoría de la información con la práctica social.

El objeto social de la UEB Centro de Preparación de Operarios "Oscar Lucero Moya", subordinada al Grupo Empresarial de la Construcción de La Habana del Ministerio de la Construcción, consiste en garantizar la formación y el perfeccionamiento -mediante cursos y diplomados en oficios básicos y especialidades- de obreros calificados, técnicos, directivos y profesionales del sector de la construcción en la capital. Su actividad fundamental es docente.

En la estructura organizativa de este centro, los departamentos de Personal, Subdirección Técnica – Docente, Enseñanza Práctica, Investigaciones, Actividad de Postgrado y Administración que agrupa las áreas Contabilidad, Finanzas, Recursos Humanos, Servicios y Almacén reciben, procesan y generan información de interés para la Secretaría Docente. Todas las áreas se subordinan a la Dirección.

Cada una de estas áreas carece de software orientado a sus actividades, no obstante, las estaciones de trabajo existentes no son suficientes y además tienen bajas prestaciones. La secretaria realiza la totalidad de las actividades de forma manual o mecánica, y ocasionalmente, con el empleo de tiempo de máquina

en otras áreas, lo que trae consigo insuficiencias tales como: demora en la elaboración de documentos, control engorroso de registros y expedientes, retraso en la búsqueda, análisis y la actualización de la información y propensa a errores, demora en la toma de decisiones que dependen del flujo informativo y no retroalimentación departamental, débil mercadotecnia y elevado costo de operación.

El proceso de perfeccionamiento empresarial que exige el Ministerio de la Construcción en sus entidades de subordinación local, obliga a esta UEB a mejorar el tratamiento de la información, pues las proyecciones de trabajo contemplan el incremento de la cantidad de cursos, la ampliación del mercado y elevar la calidad de los servicios, objetivos que en las condiciones actuales no podrían alcanzarse a corto y mediano plazo.

Por lo anteriormente expuesto se define como **problema a resolver** ¿Cómo contribuir a la mejora de la gestión de información académica en el Departamento Secretaría Docente de la UEB Centro de Preparación de Operarios "Oscar Lucero Moya"?

El **objeto de estudio** se centra en los sistemas de gestión de información y el **campo de acción** se delimita a los sistemas de gestión de información académica.

Es **objetivo general** desarrollar una solución informática que contribuya a la mejora de la gestión de información académica en el Departamento Secretaría Docente de la UEB Centro de Preparación de Operarios "Oscar Lucero Moya" para perfeccionar el proceso docente –educativo.

Se **defiende la idea** que si se desarrolla una solución informática que contribuya a la mejora de la gestión de información académica en el Departamento Secretaría Docente de la UEB Centro de Preparación de Operarios "Oscar Lucero Moya" se perfeccionará el proceso docente –educativo.

Para dar cumplimiento al objetivo general, se propone la realización de los siguientes **objetivos específicos:**

1. Elaborar el marco teórico –conceptual referente a los sistemas de gestión de información académica.
2. Realizar el análisis y diseño de la solución informática propuesta para el Departamento Secretaría Docente de la UEB Centro de Preparación de Operarios “Oscar Lucero Moya”.
3. Realizar la implementación de la solución informática propuesta para el Departamento Secretaría

Docente de la UEB Centro de Preparación de Operarios "Oscar Lucero Moya".

4. Realizar pruebas a la solución informática obtenida para el Departamento Secretaría Docente de la UEB Centro de Preparación de Operarios "Oscar Lucero Moya".

Para dar cumplimiento a los objetivos específicos se han trazado las siguientes **tareas de investigación:**

1. Caracterización de sistemas de gestión de información académica para la elaboración del marco teórico –conceptual de la investigación.
2. Caracterización del proceso de gestión de información académica en el Departamento Secretaría Docente de la UEB Centro de Preparación de Operarios "Oscar Lucero Moya".
3. Levantamiento de requisitos, para conocer las capacidades y condiciones que debe poseer el sistema.
4. Estudio de metodologías, herramientas y tecnologías para el desarrollo de la solución informática propuesta.
5. Selección de la metodología, herramientas y tecnologías para el desarrollo de la solución informática propuesta.
6. Definición del modelo conceptual referente a la gestión de información académica en el Departamento Secretaría Docente de la UEB Centro de Preparación de Operarios "Oscar Lucero Moya".
7. Realización de la exploración y planificación del sistema a desarrollar con el empleo de la metodología de desarrollo seleccionada.
8. Elaboración del diseño del sistema con el empleo de la metodología de desarrollo seleccionada y patrones de diseño.
9. Implementación del sistema informático Sistema de Gestión de Información Académica (SGIA) para dar solución a los requerimientos y contribuir a la mejora de la gestión de información académica en el Departamento Secretaría Docente de la UEB Centro de Preparación de Operarios "Oscar Lucero Moya".
10. Realización de pruebas al sistema informático "SGIA" del Departamento Secretaría Docente de la UEB Centro de Preparación de Operarios "Oscar Lucero Moya".

Partiendo del método dialéctico materialista como método general, se emplearon además métodos de orden teórico y empírico.

Métodos Teóricos.

Analítico-Sintético: se empleó para explorar los referentes teóricos sobre los sistemas de gestión de información académica, analizar los elementos conceptuales asociados al problema de investigación y extraer los que aplican al tema.

Histórico-Lógico: se empleó para constatar cómo surgieron, evolucionaron, la actualidad y tendencias de los sistemas de gestión de información académica.

Métodos Empíricos.

Entrevistas: se realizó a directivos, jefes de área y/o departamento, funcionarios, profesores y estudiantes, beneficiarios del sistema, para identificar sus necesidades y determinar las principales funcionalidades de la aplicación.

Observación: se utilizó para dar seguimiento a la evolución de la investigación, detectar dilaciones o fallas en el cumplimiento del cronograma de trabajo y reunir la información de los indicadores de las variables que figuran en el problema.

El presente trabajo de diploma está conformado por 3 capítulos, los cuales se describen a continuación:

Capítulo 1: Fundamentación teórica. Trata sobre algunos sistemas de gestión de información académica existentes tanto a nivel nacional como internacional, además se aborda sobre las metodologías de desarrollo de software, los lenguajes de programación y de modelado, así como herramientas y tecnologías utilizadas para el desarrollo de este tipo de sistema.

Capítulo 2: Exploración, planificación y diseño del sistema. Define el negocio y describe la solución propuesta para la situación problemática. Presenta las características y funcionalidades del sistema a partir de los requisitos funcionales y no funcionales capturados. Realiza el análisis, la exploración, planificación y diseño del sistema.

Capítulo 3: Implementación y pruebas de la aplicación. Expone lo referente a la implementación y las pruebas realizadas al software, para de esta manera evaluar su funcionamiento de acuerdo a las historias de usuario definidas.

Capítulo 1: Fundamentación teórica

Introducción

En este capítulo se exponen los elementos teóricos que sustentan el objetivo de la investigación, así como los elementos generales de los sistemas de gestión de información y dentro de estos, argumentos necesarios para el entendimiento de la gestión de información académica, así como el estudio de algunos sistemas desarrollados con este fin y las tendencias de desarrollo de software para la gestión de información.

1.1 Sistemas de gestión de información

Un sistema de gestión es un conjunto de etapas unidas en un proceso continuo (Ideación, Planeación, Implementación y Control), permite trabajar ordenadamente una idea hasta lograr mejoras y su continuidad. Ello engloba la información compartida, evaluaciones, trabajo en equipo y un funcionamiento acorde a principios de calidad.

Por su parte la gestión de información es: "(...) el proceso mediante el cual se obtienen, despliegan o utilizan recursos básicos (económicos, físicos, humanos, materiales) para manejar información dentro y para la sociedad a la que sirve. Tiene como elemento básico la gestión del ciclo de vida de este recurso y ocurre en cualquier organización. Es propia también de unidades especializadas que manejan este recurso en forma intensiva, llamadas unidades de información" (1).

La gestión de información tiene el objetivo de:

- Maximizar el valor y los beneficios derivados del uso de la información.
- Minimizar el costo de adquisición, procesamiento y uso de la información.
- Determinar responsabilidades para el uso efectivo, eficiente y económico de información.
- Asegurar un suministro continuo de la información (2).

Una de las clasificaciones de los sistemas de gestión de información la constituyen los sistemas de gestión de información académica, que tienen como principal objetivo proveer a las instituciones educacionales de mecanismos automatizados que faciliten la planificación, organización, gestión y control académico (3).

1.2 Sistemas de gestión de información académica

La gestión de información académica consiste en el proceso de planear, reclutar, seleccionar, capacitar, desarrollar, motivar, estimular, controlar y evaluar el desempeño de trabajadores, estudiantes de una entidad con vistas al logro de sus objetivos (4).

La gestión académica tiene su núcleo en las secretarías docentes de los centros de educación. Estas constituyen la unidad administrativa con características muy específicas, pues su misión no es otra que la de servir de hilo conductor a la gestión administrativa del alumno desde el inicio hasta la finalización de los estudios, gestionar los procesos académicos, coordinar y planificar sus actuaciones. Su importancia radica en la estrategia de formación y sus estudios estadísticos, realiza el diseño, implantación y operación de las decisiones académicas, se evalúan los programas académicos y se asegura su calidad (4).

Con el objetivo de lograr un acercamiento a las tendencias actuales sobre los sistema de gestión de la información científica académica, tanto internacionales como nacionales y valorar hasta qué punto se pudiera encontrar una solución que cumpla con las expectativas del cliente se analizaron una serie de sistemas a nivel nacional e internacional.

1.2.1 Sistemas de Gestión de información académica a nivel nacional e internacional

Sistema Docente Integral (SDI)

Este sistema se creó en el Instituto Superior Politécnico "José Antonio Echeverría" (CUJAE) y surgió debido a la vital importancia que tiene para este centro el control, planificación y resultado de los procesos docentes. El SDI permite registrar, procesar y crear mecanismos de recuperación de la información relacionada con el proceso docente, tanto de pregrado como de postgrado de forma automatizada. Esto constituye una ventaja pues con su uso se pueden realizar estos procesos de forma eficiente permitiendo optimizar los recursos del centro secretaría de facultades (5).

Este sistema dentro de los módulos implementados contemplan la recuperación de información, pagos de estipendios, módulos de postgrado pero que aún estos no se corresponden con las necesidades planteadas en el presente trabajo de realizar la gestión de diplomados y un mejor control del plan de estudio.

En los 12 años de trabajo de la UCI, se han utilizado varios sistemas de gestión académica, con sus

diferentes versiones. Inicialmente, se empleó el sistema **GESTACAD** realizado en la Universidad de Matanzas. Luego, se decidió desarrollar una solución propia basada en las experiencias obtenidas con el uso de este sistema que incluyera nuevas operaciones, el cual se nombró **UCIMAT**.

Debido a las limitaciones que presentaba el sistema de gestión académica **UCIMAT** se creó un nuevo sistema llamado **Akademos**, solución que estuvo en uso desde el 2005 e implementada con herramientas propietarias lo cual no cumplía con los planes de migración a software libre que se trazaron la UCI y el país, por lo que se hizo necesario implementar un nuevo sistema, más adecuado a la universidad, el Sistema de Gestión Universitaria (6).

Sistema de Gestión Universitaria

Sistema utilizado actualmente para la gestión académica. Es una solución integral para la gestión de los procesos sustantivos de la universidad. Cuenta con varios módulos, entre ellos el de Pregrado. Este módulo permite la gestión de más de una carrera, la gestión de bajas, traslados y licencias, además es la herramienta utilizada para la gestión de los procesos de formación de los estudiantes, así como la gestión de personal, registro y control docente de las actividades de formación. Sin embargo, como consecuencia de los cambios realizados en los planes de estudio y en el proceso de formación de sus educandos, han surgido nuevas necesidades referentes a la gestión de la formación docente de sus estudiantes que no están presentes en este sistema. Algunas de ellas son la gestión de los ajustes docentes, imprescindibles en caso de repitencia, arrastre o reincorporación, lo que dificulta la asignación de los estudiantes a los grupos docentes y hace el trabajo del personal de Secretaría Docente y de los vicedecanos de formación engorrosa y propensa a errores. El sistema no gestiona los trabajos de diploma de los estudiantes, lo que dificulta la asignación de tutores de tesis, la creación de los comités y tribunales de tesis, además de no registrar las evaluaciones ni observaciones de los talleres de tesis correspondientes.

UNAN-León

Sistema académico de la Universidad Nacional Autónoma de Nicaragua de León, su objetivo es proveer a la universidad de mecanismos automatizados que faciliten la planificación, organización, gestión y control académico (7).

Entre las principales funciones del sistema tenemos:

- Simplificar y organizar los trámites y procesos académicos.

- Apoyar a las secretarias académicas en los procesos académicos.
- Registrar y controlar las pre-matrículas.
- Registrar y controlar las matrículas.
- Registrar y controlar el pago de aranceles de los estudiantes.
- Registrar y controlar las becas.
- Registrar y controlar las estadísticas.
- Registrar y controlar los convenios de la universidad.
- Emitir resultados del proceso de admisión.

UNAN-León no contempla la planificación de diplomados. Usa tecnología propietaria, por lo que su adopción iría contra las políticas de migración a software libre llevadas a cabo en el país. Está orientado a organizar los trámites y procesos académicos, registrar y controlar las matrículas y pre-matrículas de los estudiantes; por lo que no se adapta a los procesos académicos de la UEB Centro de Preparación de Operarios "Oscar Lucero Moya". Así como no se posee la documentación de su desarrollo en función de emplear la misma tecnología de software y demás herramientas.

Sistema de Información de Gestión Académica (SIGA)

Es un producto puesto en marcha en la Universidad de Chile que permite recopilar y tratar la información de los distintos niveles institucionales, lo que relaciona la labor docente, investigativa, de creación artística y extensión, permitiendo conocer cuantitativa y cualitativamente las iniciativas desarrolladas por los distintos organismos de la Institución.

A partir de los datos e información ingresados por las diversas unidades de la institución, es posible generar indicadores y reportes de análisis en docencia de pregrado, postgrado, de actividad académica, de investigación y extensión.

De las funcionalidades encontradas en este software podemos mencionar:

- Presenta módulos como: personal académico y actividad académica donde trata todo lo referente a docencia y a actividades tanto de extensión como difusión.
- Permite de forma automática controlar la asistencia del estudiante, estadísticas y porcentaje. Genera listados para los expedientes académicos, calificaciones de un alumno/curso, observaciones, diplomas, alta, baja, consulta y modificación de profesores. (8).

SIGA no contempla la planificación de diplomados. Usa tecnología propietaria, por lo que su adopción iría contra las políticas de migración a software libre llevadas a cabo en el país. Permite al estudiante preinscribir/inscribir asignaturas durante el proceso de matrícula de cada periodo académico. Contiene información sobre las asignaturas cursadas, notas, reprobaciones, convalidaciones y asignaturas cursadas fuera del plan. Por lo que no se adapta a los procesos académicos de la UEB Centro de Preparación de Operarios "Oscar Lucero Moya". Así como no se posee la documentación de su desarrollo en función de emplear la misma tecnología de software y demás herramientas.

1.3 Metodologías de desarrollo de software

Una metodología de desarrollo en la ingeniería de software es un conjunto de procedimientos, técnicas, herramientas y soporte documental que debe seguirse para el desarrollo del software, proporcionando una ayuda muy importante e indispensable para que el producto final posea las funcionalidades requeridas por el cliente y que cumpla con las necesidades del mismo y del usuario final (9).

Se destacan dos tipos de metodologías teniendo en cuenta su filosofía, aquellas con mayor énfasis en la planificación y control del proyecto, en especificación precisa de requisitos y modelado, reciben el apelativo de Metodologías Tradicionales o Pesadas. Dentro de ellas se puede hacer referencia a: Rational Unified Procces (RUP), Microsoft Solution Framework (MSF), Win-Win Spiral Model, Iconix.

Las que por su parte se adaptan al cambio teniendo un tiempo de respuesta rápido son denominadas Metodologías Ágiles, las mismas tienen como base los procesos ágiles, las cuales son una buena elección cuando se trabaja con requisitos desconocidos o no son estables y por tanto no se puede seguir un proceso totalmente planificado. Dentro de éstas se encuentran: Extreme Programming (XP), SCRUM, Crystal Clear, Adaptive Software Development (ASD), XBreed, SXP (10).

Proceso Unificado de Desarrollo (RUP)

Proceso Unificado de Desarrollo (RUP por sus siglas en inglés), este posee tres aspectos que la definen:

- Centrado en la arquitectura: Los arquitectos deben diseñar el sistema de forma tal que el sistema evolucione, no solo durante la etapa inicial sino también en las generaciones venideras.
- Iterativo e incremental: En los sistemas grandes es práctico dividir el trabajo en partes más pequeñas o mini-proyectos, donde cada uno es una iteración que posteriormente se convierte en un incremento o crecimiento del producto.

RUP define disciplinas o flujos a realizar en cada fase del proyecto: modelado del negocio, análisis de requisitos, análisis y diseño, implementación, pruebas, distribución o despliegue, gestión de configuración y cambios, gestión del proyecto y gestión del entorno. Este posee gran cantidad de artefactos a generar, además posee gran cantidad de roles lo cual no lo hace efectivo para equipos de trabajo pequeños, de requisitos cambiantes y que necesite entregarse en poco tiempo (12).

Extreme Programming (XP)

La metodología XP consiste básicamente en ajustarse estrictamente a una serie de reglas que se centran en las necesidades del cliente para lograr un producto de buena calidad en poco tiempo. Es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de software. Promueve el trabajo en equipo, preocupándose en todo momento del aprendizaje de los desarrolladores y estableciendo un buen clima de trabajo (13).

Esta metodología se basa en una realimentación continuada entre el cliente y el equipo de desarrollo con una comunicación fluida entre todos los participantes, también busca simplificar las soluciones implementadas y coraje para los múltiples cambios. Este tipo de programación es la adecuada para los proyectos con requisitos imprecisos y muy cambiantes.

Los principios y prácticas son de sentido común pero llevadas al extremo, de ahí proviene su nombre. Kent Beck, el padre de XP, describe la filosofía de XP en dos: sin cubrir los detalles técnicos y de implantación de las prácticas.

Las prácticas fundamentales de la metodología XP son:

- **Entregas pequeñas:** La idea es producir rápidamente versiones del sistema que sean operativas, aunque no cuenten con toda la funcionalidad pretendida para el mismo.
- **Refactorización:** La refactorización es una actividad constante de reestructuración del código con el objetivo de remover duplicación de código, mejorar su legibilidad, simplificarlo y hacerlo más flexible para facilitar los posteriores cambios. También mejora la estructura interna del código sin alterar su comportamiento externo.
- **Propiedad colectiva del código:** Cualquier programador puede cambiar una parte del código en un momento dado. Esta práctica motiva a todos a contribuir con nuevas ideas en todos los

segmentos del sistema, evitando a la vez que algún programador sea imprescindible para realizar cambios en alguna porción de código (16).

La **Figura 1** muestra como esta metodología es iterativa y utiliza pequeños lapsos de tiempo para realizar las actividades.

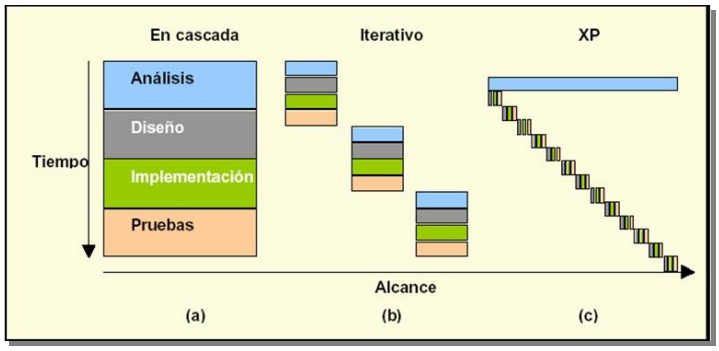


Figura 1. Comparación de los ciclos de desarrollo en cascada, iterativos y XP (14).

Metodología a utilizar

Se selecciona XP, pues es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de software, promueve el trabajo en equipo, se preocupa por el aprendizaje de los desarrolladores y propicia un buen clima de trabajo. Se basa en la retroalimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. Es ideal para proyectos donde el ciclo de vida es relativamente corto por lo que puede adaptarse a la solución propuesta, los cambios a los requerimientos son bienvenidos, aún en fases tardías del desarrollo, se centra fundamentalmente en entregar un producto de software en el menor tiempo posible y con calidad. Se define como: especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes.

1.4 Tecnologías para el desarrollo de Sistemas de Gestión web

1.4.1 Lenguaje de modelado

UML (Lenguaje unificado de modelado)

Es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el OMG (Grupo de Gestión de Objetos). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software. UML ofrece un estándar para describir un plano del

sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables (18).

Modelado de proceso de negocio

BPMN (Notación de Modelado de Procesos de Negocio) es utilizada como un estándar para la representación de procesos de negocios, independientemente de la metodología empleada para automatizar dichos procesos. Esta notación ha sido diseñada para coordinar la secuencia de los procesos y los mensajes que fluyen entre los participantes en las actividades. Proporciona un lenguaje común para que las partes involucradas en el análisis de negocios (cliente y desarrollador) puedan intercambiar sobre un mismo modelo de forma clara, precisa y eficiente (19).

Se selecciona UML como lenguaje de modelado para especificar, visualizar, construir y documentar los artefactos del sistema. Es un lenguaje estándar, fácil de aprender, y ofrece una amplia variedad de diagramas para mostrar el sistema desde varias perspectivas. Está especialmente diseñado para apoyar un estilo de desarrollo iterativo e incremental y presenta tecnología orientada a objetos. Tiene una notación gráfica muy expresiva que permite representar en mayor o menor medida todas las fases de un proyecto informático: desde el diseño con los diagramas de clases, objetos, hasta la implementación y configuración con los diagramas de despliegue.

1.4.2 Herramientas CASE (Ingeniería de software asistida por computadora)

Las herramientas de Ingeniería de Software Asistido por Computadora (CASE, en inglés Computer Aided Software Engineering) son aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de un software reduciendo el costo del mismo. Pueden servir de ayuda durante el ciclo de vida de desarrollo del software en tareas como el diseño del proyecto, cálculo de costos, implementación de parte del código automáticamente a partir del diseño dado, compilación automática, documentación o detección de errores. Existen múltiples herramientas con estos fines, tales como Enterprise Architect, Visual Paradigm, ArgoUML, Rational Rose, entre otras.

Visual Paradigm

Es una herramienta CASE que propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasa por el análisis y el diseño, hasta la generación del código fuente

de los programas y la documentación. Soporta el ciclo de vida completo del proceso de desarrollo de software a través de la representación de todo tipo de diagramas. Fue diseñada para una amplia gama de usuarios interesados en la construcción de sistemas de software de forma fiable a través de la utilización de un enfoque orientado a objetos.

Algunas de sus características son:

- Disponibilidad en múltiples plataformas.
- Diseño centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad.
- Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- Capacidades de ingeniería directa e inversa.
- Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
- Disponibilidad de múltiples versiones, para cada necesidad.
- Licencia: gratuita y comercial.
- Soporta aplicaciones Web.

Esta herramienta permite aumentar la calidad del software a través de la mejora de la productividad en el desarrollo y mantenimiento del software. Aumenta el conocimiento informático de una empresa y ayuda a la búsqueda de soluciones para los requisitos. También permite la reutilización del software, portabilidad y estandarización de la documentación, además del uso de las distintas metodologías propias de la Ingeniería del Software (20).

Rational Rose

Es una herramienta CASE, desarrollada por Rational Corporation basada en UML, que permite crear los diagramas que se generan durante el proceso de ingeniería en el desarrollo del software.

Características adicionales:

- Característica de control por separado de componentes modelo que permite una administración más granular y el uso de modelos.
- La generación de código Ada, ANSI C ++, C++, CORBA, Java y Visual Basic, con capacidad de sincronización modelo –código configurables.
- Capacidad de análisis de calidad de código.
- Publicación web y generación de informes para optimizar la comunicación dentro del equipo (21).

Se selecciona la herramienta Visual Paradigm que permite la interoperabilidad entre diagramas, es capaz de exportar los diagramas de un modelo a otro con facilidad, además de permitir la transformación de diagramas de Entidad-Relación en tablas de base de datos, también realiza ingeniería inversa a bases de datos desde sistemas gestores de bases de datos existentes a diagramas de Entidad- Relación.

1.4.3 Lenguajes de Programación

Es aquel elemento dentro de la Informática que nos permite crear programas mediante un conjunto de instrucciones, operadores y reglas de sintaxis que se ponen a disposición del programador para que este pueda comunicarse con los dispositivos de hardware y software existentes. Estos se componen de un conjunto de reglas sintácticas y semánticas que permiten expresar instrucciones que luego serán interpretadas (22).

Las aplicaciones web para internet e intranet presentan una serie de ventajas y beneficios con respecto al software de escritorio, con lo cual logra aprovechar y acoplar los recursos de su empresa de una forma mucho más práctica que el software tradicional (23). En la actualidad existen diferentes lenguajes para el desarrollo web. PHP es el de mayor nivel comercial, aunque no se puede descartar a Java como otro de los lenguajes de gran utilización y preferencia (24).

Lenguajes de desarrollo del lado del cliente

Hyper Text Markup Language 5 (HTML5)

Este es la última de cinco revisiones, es una colección de estándares para el diseño y desarrollo de páginas web. Esta colección representa la manera en que se presenta la información en el explorador de internet y la manera de interactuar con ella (25). Provee básicamente tres características: estructura, estilo y funcionalidad. Es considerado el producto de la combinación de HTML, CSS y JavaScript. Estas tecnologías son altamente dependientes y actúan como una sola unidad organizada bajo la especificación de HTML5. Esta nueva versión ofrece ventajas: surgen varios elementos que sirven para estructurar mejor una página web, mejoras en los formularios, aumenta la cantidad de tipos de datos que soporta, agrega nuevas etiquetas y elimina aquellas ineficientes o que tendían a confundir a los desarrolladores.

CSS

Es un lenguaje de hojas de estilos creado para controlar la presentación de los documentos electrónicos definidos con HTML y XHTML, es la forma de separar los contenidos y su presentación y es

imprescindible para la creación de páginas web complejas dándole un nivel de organización que permite el desarrollo de estas (25).

La versión **3** de **CSS** es la que se utilizará debido a su fácil acoplamiento con HTML5, a continuación se muestra una serie de propiedades que describe: backgrounds y borders entre las principales características, destacan que se puede agregar múltiples backgrounds a un objeto, los borders pueden ser redondos con posibilidad de indicar el radio de curvatura; multi-column layout simplifica el proceso de crear diseños con múltiples columnas sin etiquetas adicionales; advanced layout nueva característica que permite organizar los elementos en pantalla de una mejor manera y combinarlos de diferentes maneras sin etiquetas adicionales; selectors permite agregar efectos a acciones que se realicen, las posibilidades para el uso de selector son realmente muchas -permite seleccionar un elemento web para darle estilo o estructura (25).

Ventajas de CSS (26):

- Conserva el ancho de banda del usuario, lo que acelera la carga de páginas, especialmente en conexiones telefónicas.
- Reduce la sobrecarga del servidor.
- Reduce el tiempo de diseño y programación.
- Aumenta la accesibilidad al eliminar algunos, muchos o todos los elementos de presentación del marcado.

JavaScript

Lenguaje de programación que se utiliza para crear páginas web dinámicas. Permite incorporar efectos como aparición y desaparición de texto, animaciones, acciones que se activan al pulsar botones u otros elementos y ventanas con mensajes de aviso al usuario. JavaScript es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos; con el uso de JavaScript se puede probar directamente en cualquier navegador el código, sin necesidad de procesos intermedios (25).

Entre las principales ventajas que posee este lenguaje de programación se encuentran (27):

- Fácil de aprender, rápido y potente: como este lenguaje es muy sencillo de aprender se puede empezar a trabajar con él desde el principio. Es ideal para agregar ciertas funciones rápidas a una

página web.

- Usabilidad: JavaScript es, con diferencia, el lenguaje de programación que más se utiliza en la web.
- Hay publicadas millones de páginas web que incorporan elementos que lo usan. La mayoría de los navegadores web pueden trabajar con él.
- Reducción de la carga del servidor: este lenguaje se puede hacer cargo de una gran parte de las funciones del cliente de las cuáles se encargaba el servidor, siendo un ejemplo de esto las validaciones.

Lenguaje de desarrollo del lado del servidor

Hipertext Pre-processor (PHP)

Es un lenguaje del lado del servidor (esto significa que PHP funciona en un servidor remoto que procesa la página web antes de que sea abierta por el navegador del usuario). Combinado con la base de datos MySQL, es el lenguaje estándar a la hora de crear sitios de comercio electrónico o páginas web dinámicas. Puede ser incluido con facilidad dentro del código HTML, y permite una cantidad enorme de funcionalidades que se ha convertido en el favorito de millones de programadores en todo el mundo. (28).

Entre sus características fundamentales están:

- Gratuito: al tratarse de software libre, puede descargarse y utilizarse en cualquier aplicación personal o profesional, de manera completamente libre.
- Gran popularidad: existe una gran comunidad de desarrolladores y programadores que continuamente implementan mejoras en su código.
- Con escaso mantenimiento y un servidor gratuito, puede soportar sin problema millones de visitas diarias.
- Sencilla integración con múltiples bases de datos, MySQL es la base de datos que mejor trabaja con PHP, puede conectarse también a PostgreSQL, Oracle o cualquier otra base de datos compatible con el conector ODBC.
- Versatilidad. PHP puede usarse con la mayoría de los sistemas operativos, ya sea basados en UNIX (Linux, FreeBSD), como con Windows, el sistema operativo de Microsoft (28).

Se decide utilizar PHP 5 por ser un lenguaje libre, el desarrollador del sistema posee amplio conocimiento del mismo, es completamente orientado al desarrollo de aplicaciones web dinámicas con acceso a

información almacenada en una base de datos, presenta gran compatibilidad con MySQL, existe gran cantidad de documentación y cuenta con una biblioteca de funciones sumamente amplia.

1.4.4 Framework (Marco de trabajo)

El concepto framework se emplea en muchos ámbitos del desarrollo del software. En general, se refiere a una estructura de software compuesta de componentes personalizables e intercambiables para el desarrollo de una aplicación (29). Se puede ver como una aplicación genérica incompleta y configurable a la que se le puede añadir las últimas piezas para construir una aplicación funcional (30).

Los objetivos principales que persigue el trabajo con framework según (31) son: acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones, lo cual se ajusta a las necesidades del problema que guía esta investigación y por lo que se concibe su uso como parte de la solución.

Framework de desarrollo del lado del cliente

Framework de CSS

Dentro de los framework de CSS se encuentran: Blueprint, Less Framework, Baseline, YUI Grids CSS, 960 Grid System, Bootstrap, entre otros. Específicamente Bootstrap es un paquete de herramientas que simplifica el proceso de creación de diseños web mediante la combinación de CSS y JavaScript. Ofrece una serie de plantillas CSS y ficheros JavaScript que permiten integrar el framework de forma sencilla y potente en proyectos web. Permite crear interfaces que se adapten a los diferentes navegadores, tanto de escritorio, como tablets y móviles a distintas escalas y resoluciones. Se integra perfectamente con las principales librerías JavaScript. Es ligero y funciona con todos los navegadores. Ofrece un diseño sólido, así como distintos componentes que pueden utilizarse con unos estilos predefinidos y fáciles de configurar. Se selecciona Bootstrap para maquetar la solución que se propone, debido a las características antes mencionadas, también por ser el framework el del autor de un mayor dominio de su aplicación.

Framework de JavaScript

Dentro de los framework de JavaScript se puede citar: JQuery, MooTools, Prototype, entre otros que pueden ser consultados en el sitio BestWebFramework. Específicamente JQuery: es una biblioteca de JavaScript, creada por John Resig, que permite simplificar la manera de interactuar con los documentos

HTML, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX la cual será utilizada debido a su fácil integración con Symfony.

JQuery 2 fue el framework escogido por ser fácil de usar y aprender, posee gran documentación y cuenta con un gran equipo de desarrolladores a cargo de la mejora, actualización del framework y creación de plugins o componentes facilitando encontrar soluciones ya creadas para implementar asuntos como interfaces de usuario y galerías. Es de software libre y de código abierto, posee un doble licenciamiento bajo la licencia MIT y la Licencia Pública General de GNU v2, permite su uso en proyectos libres y privativos. JQuery, al igual que otras bibliotecas, ofrece una serie de funcionalidades basadas en JavaScript que de otra manera requerirían de mucho más código.

Framework de desarrollo del lado del servidor

Los Framework PHP han recibido mucha atención en los últimos años de parte de la comunidad de desarrolladores web. Estos se caracterizan por la organización de código y de archivos, poseen una estructura en sus carpetas predeterminadas mediante estándares y permitiendo la organización del código. Las facilidades brindadas por las utilidades y librerías ayudan a los procesos de validación de formularios, filtración de datos de entrada/salida, abstracción para bases de datos y utilidades para manejar cookies, sesiones, e-mail, calendario, paginación, entre otras.

Igualmente cuentan con una amplia cantidad de plugins dados por la comunidad que se pueden añadir. La utilización del patrón Modelo Vista Controlador (MVC) permite la separación del código, haciéndolo más limpio y fácil de mantener. Son muy seguros pues existen muchas funciones para filtrar datos de entrada y de salida para protegerse de ataques. La estructura que posee por defecto hace más rápido y eficiente el proceso de detectar bugs, mantiene el código limpio y hace cambios de manera sencilla, además, los principales framework de PHP tienen grandes y activas comunidades que permiten la comunicación con otros desarrolladores, así como obtener ayuda y feedbacks (32).

Es de gran beneficio usar un framework PHP para proyectos con fecha de entrega ajustada o proyectos con una cantidad importante de código similar, lo que permite ahorrar tiempo y ajustar desperfectos en la codificación. Entre los framework PHP más conocidos se encuentran según (33): Zend Framework, Yii, CodeIgniter y Symfony2.

Zend Framework: es un framework de código abierto para desarrollar aplicaciones web y servicios web con PHP 5. Su implementación está orientada a objetos. Posee una estructura donde cada componente está construido con una baja dependencia de otros componentes permitiéndoles trabajarlos por separado, está enfocado a las aplicaciones tipo web 2.0; tiene propiedades importantes que están construidas para el desarrollo a nivel corporativo y requiere un gran conocimiento de PHP por parte del desarrollador (34).

CodeIgniter: es un framework para aplicaciones web de código abierto para crear sitios web dinámicos muy conocido por su utilización simple, su desempeño y velocidad. Pero no utiliza las últimas tecnologías PHP, como namespace u ORM (Object-Relational Mapping) aunque este último se logra con la integración de librerías (35).

Symfony2: es un framework completo con componentes bien integrados. Integra una capa ORM como Doctrine o Propel (36). Está más enfocado en el desarrollo de aplicaciones de diferentes tipos de niveles de complejidad. Es estable y reconocido, con una comunidad activa de la cual se puede obtener información y ayuda. Varios proyectos y empresas importantes a nivel mundial lo utilizan como por ejemplo: Yahoo, Dailymotion, Opensky, Drupal, Behat, Doctrine, Propel, PHP Unit, Jackalope, Silex, PPI, Easybook, Midgard CMS, phpBB (37).

Después de un minucioso estudio de los frameworks se seleccionó Symfony2 como framework de desarrollo para el sistema a desarrollar. El anexo (Anexo 1), muestra los resultados obtenidos de por qué se seleccionó Symfony para el desarrollo de este trabajo de diploma.

Framework de desarrollo de PHP Symfony2

Symfony2 en su versión 2.6.1 tiene una arquitectura interna completamente desacoplada, lo que permite reemplazar o eliminar fácilmente aquellas partes que no son necesarias para un proyecto (37). Como características principales de este framework según (38) se destaca por ser fácil de instalar y configurar en la mayoría de plataformas, independiente del sistema gestor de bases de datos, sencillo de usar en la mayoría de los casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos, sigue la mayoría de las mejores prácticas y patrones de diseño para la web, preparado para aplicaciones empresariales, adaptable a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo, fácil de extender lo que permite su integración con librerías desarrolladas por terceros.

Symfony2 utiliza **YAML** (Ain't Another Markup Language) para realizar los archivos de configuración, aunque también es compatible con XML, PHP y anotaciones nativas. Los diferentes formatos son compatibles y se pueden utilizar indistintamente en una aplicación.

Entre las ventajas de utilizar YAML para realizar los archivos de configuración se pueden mencionar:

- Encontrar el balance adecuado entre características y velocidad.
- El analizador es bastante robusto, fácil de entender, y lo suficientemente simple como para ampliarlo (39), (40).
- Como herramienta fundamental, utiliza el motor Twig para generar plantillas, aunque también puede utilizar las nativas de PHP incluso la combinación de ambas (39).

Otro elemento importante es el uso de las bases de datos, que siguen en su mayoría una estructura relacional y Symfony2 por el contrario está orientado a objetos. Por este motivo, para acceder a la base de datos como si fuera orientada a objetos, es necesaria una interfaz que traduzca la lógica de los objetos a la lógica relacional. Esta interfaz se denomina mapeo de objetos a bases de datos o Object-Relational Mapping **ORM**, Symfony2 es compatible con dos, **Propel** y **Doctrine** es este último el que trae por defecto (38).

Tabla 1. Comparación entre los ORM Doctrine y Propel. (Elaboración Propia) según análisis de (36), (41) y (42).

ORM	Doctrine	Propel
Operaciones CRUD (Create, Retrieve, Update and Delete)	Si	Si
Generación de clases PHP	YAML o PHP	XML
Soporta múltiples motores	Si	Si
Herencia	Simple y de agregación	Simple
Behaviors	Si	No
Motor de búsqueda	Si	No
Documentación	Alta	Media

Ambos **ORM** brindan facilidades al interactuar con las bases de datos, pero después de analizar la Tabla

1, se concluye que Doctrine tiene excelente documentación, mejores características y además es el ORM con el que por defecto trabaja Symfony2. Su organización interna está basada en "bundle". Un "bundle", contiene una estructura, cuenta con los controladores que se encargan de la lógica de negocio, las vistas visualizadas por los usuarios y el modelo de datos. Estos "bundle" les dan la flexibilidad de usar componentes de terceros, están desarrollados de modo sencillo para elegirlos y habilitarlos en la aplicación (43).

1.4.5 IDE (Entorno integrado de desarrollo)

Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica, lo cual proporciona un marco de trabajo amigable para la mayoría de los lenguajes de programación tales como C++, Python, Java, PHP, C#, Delphi y Visual Basic. Los IDE pueden ser aplicaciones por sí solas o pueden ser parte de aplicaciones existentes (44).

NetBeans

Es una herramienta para programadores pensada para escribir, compilar, depurar y ejecutar programas; está escrito en Java, pero puede servir para cualquier otro lenguaje de programación; existe además un número importante de módulos para extender el IDE NetBeans; es un producto libre y gratuito sin restricciones de uso.

Esta plataforma es una base modular y extensible usada como una estructura de integración para crear aplicaciones de escritorio grandes. Existen empresas independientes asociadas y especializadas en desarrollo de software, que proporcionan extensiones adicionales, las cuales se integran fácilmente en la plataforma y pueden también utilizarse para desarrollar sus propias herramientas y soluciones.

Entre las características de la plataforma están:

- Administración de las interfaces de usuario (ej. menús y barras de herramientas).
- Administración de las configuraciones del usuario.
- Administración del almacenamiento (guarda y carga cualquier tipo de dato).
- Administración de ventanas.
- Framework basado en asistentes (45).

Se seleccionó NetBeans versión 8.0 porque ofrece un excelente entorno y proporciona un rendimiento superior para los desarrolladores de PHP, además de aportar editores y herramientas integrales para sus tecnologías relacionadas. Este IDE es extensible pues permite a través de plugings; editar código escrito en los lenguajes XML, HTML, PHP y Java Script. Posee un gran soporte para la edición en HTML5, y tiene un excelente balance entre una interfaz con múltiples opciones así como un aceptable completamiento de código (23).

1.4.6 Servidores Web

Un servidor web es un programa que atiende y responde a las diversas peticiones de los navegadores; proporcionándoles los recursos que solicitan mediante el protocolo HTTP o el protocolo HTTPS (la versión segura, cifrada y autenticada de HTTP) (47).

Apache 2.4.4

Es un servidor web de código libre robusto; cuya implementación se realiza de forma colaborativa, con prestaciones y funcionalidades equivalentes a las de los servidores comerciales. El proyecto está dirigido y controlado por un grupo de voluntarios de todo el mundo que, planifican y desarrollan el servidor y la documentación a través del uso de la Internet.

Ventajas y aceptación (48):

- Modular.
- Multi-plataforma.
- Extensible.
- Popular (fácil conseguir ayuda/soporte).

Apache tiene amplia aceptación en la red: desde 1996, Apache, es el servidor HTTP más usado. Alcanzó su máxima cuota de mercado en 2005 siendo el servidor empleado en el 70% de los sitios web en el mundo.

IIS (Internet Information Services)

Es un potente servidor web que ofrece una infraestructura de gran fiabilidad, capacidad de manejo y escalabilidad para aplicaciones web sobre todas las versiones de Windows Server 2003. IIS 6.0 soporta la Iniciativa de Sistemas Dinámicos de Microsoft (DSI) con aislamiento de procesos y capacidades de

gestión mejoradas (49).

Se empleará en la solución el servidor web Apache por las características antes mencionadas, no se usará IIS porque es de la empresa Microsoft Corporation y privativo, lo que constituye una gran limitante para el desarrollo de la solución.

1.4.7 Gestores de Base de Datos

Un Sistema Gestor de Base de Datos (SGBD, en inglés DBMS: DataBase Management System) es un sistema de software que permite la definición de bases de datos, así como la elección de las estructuras de datos necesarios para el almacenamiento y búsqueda de los datos, ya sea de forma interactiva o a través de un lenguaje de programación (50).

MySQL 5.6.12

Es un sistema gestor de bases de datos relacionales, rápido, sólido y flexible. Es la base de datos más popular e idóneo para la creación de bases de datos con acceso desde páginas web dinámicas, así como para la creación de cualquier otra solución que implique el almacenamiento de datos, posibilita realizar múltiples y rápidas consultas. Está desarrollado en C y C++; facilita su integración en otras aplicaciones desarrolladas también en esos lenguajes. Es un sistema cliente/servidor, por lo que permite trabajar como servidor multiusuario y de subprocesamiento múltiple, o sea, cada vez que se crea una conexión con el servidor, el programa servidor establece un proceso para manejar la solicitud del cliente, controla así el acceso simultáneo de un gran número de usuarios a los datos y asegura el acceso solo a usuarios autorizados. Es uno de los sistemas gestores de bases de datos más utilizados en la actualidad, lo que incluye a grandes corporaciones como Yahoo!, Finance, Google, Motorola, entre otras. Se suele trabajar en combinación con PHP, y comparte con este algunas de las características que lo convierten en una elección segura (51).

Entre sus características podemos mencionar:

Puede ser utilizado sin limitación alguna y se encuentra bajo licencia GPL. Es muy popular, pues son innumerables las páginas donde encontrar información, la velocidad de proceso de MySQL es legendaria. Versatilidad: trabaja tanto con sistemas operativos basados en Unix como con el sistema operativo Windows, de Microsoft. Al utilizar el lenguaje estándar SQL es muy sencillo de manejar y el tener conocimientos de otras bases de datos nos ayudará enormemente (50).

PostgreSQL

Es un Sistema Gestor de Bases de Datos Relacionales Orientadas a Objetos, de código abierto, brinda un control de concurrencia multi-versión que permite trabajar con grandes volúmenes de datos; soporta gran parte de la sintaxis SQL y cuenta con un extenso grupo de enlaces con lenguajes de programación.

A continuación se describen alguna de sus ventajas:

- Posee características significativas del motor de datos, entre las que se pueden incluir las subconsultas, los valores por defecto, las restricciones a valores en los campos (constraints) y los disparadores (triggers).
- Ofrece funcionalidades en línea con el estándar SQL92, incluye claves primarias, identificadores entrecomillados, conversión de tipos y entrada de enteros binarios y hexadecimales.
- El código fuente se encuentra disponible para todos sin costo alguno. Disponible para 34 plataformas con la última versión estable.

Después de realizar un estudio comparativo entre los dos gestores de base de datos más reconocidos se seleccionó MySQL. Pues posee mayor velocidad al realizar las operaciones, convirtiéndolo en uno de los gestores que ofrece mayor rendimiento, su bajo consumo lo hace idóneo para ser ejecutado en una máquina con bajas prestaciones sin ningún problema, las utilidades de administración de este gestor son de gran ayuda para muchos de los gestores comerciales existentes, debido a su gran facilidad de configuración.

Conclusiones del capítulo

El análisis bibliográfico hasta la actualidad permitió comprender que es necesario el uso de los sistemas de gestión, debido al avance de las tecnologías de la información y las comunicaciones en el mundo. Los sistemas de gestión académica estudiados no constituyen una alternativa viable de solución a la problemática que dio origen a esta investigación, por lo que se descarta su empleo. Se evidencia que la mejor opción para la gestión de información académica en el Departamento Secretaría Docente de la UEB Centro de Preparación de Operarios "Oscar Lucero Moya", es desarrollar un sistema en base a la satisfacción de las necesidades específicas de dicha institución.

Para el desarrollo de la aplicación web se analizaron las metodologías de desarrollo de software más utilizadas. Como metodología de desarrollo se seleccionó a XP debido a la corta duración del proyecto para desarrollar el sistema informático. Se realizó un estudio de las principales tecnologías y herramientas llegando a seleccionar las más adecuadas para el desarrollo del sistema. Entre las seleccionadas se encuentran: como herramienta CASE Visual Paradigm 8.0, como lenguaje de modelado: UML, como servidor web: Apache 2.4.4, como IDE de desarrollo: NetBeans 8.0, como gestor de base de datos MySQL: 5.6.12. Como lenguaje de desarrollo del lado del cliente se selecciona: HTML 5, CCS 3 y JavaScript, como lenguaje de desarrollo del lado del servidor: PHP 5 y los frameworks de desarrollo: Bootstrap 2.3.2 como framework de CSS, JavaScript como framework de JQuery 2 y Symfony 2.6.1 como framework de PHP.

Capítulo 2: Exploración, planificación y diseño del sistema

Introducción

El presente capítulo tiene el modelamiento de la información referente al Departamento Secretaría Docente de la UEB Centro de Preparación de Operarios "Oscar Lucero Moya", consiste en la definición de los objetivos del sitio, especificándose los requisitos funcionales y no funcionales que debe cumplir, así como la estructura y diseño del mismo. También se hará alusión a las fases de exploración, planificación y desarrollo, las primeras de XP, el objetivo de estas fases es conocer el alcance del producto a desarrollar, así como estimar los tiempos de entrega de cada versión. En este capítulo se exponen, además, los artefactos que se generan a partir de los requerimientos expuestos por el cliente.

2.1 Modelo Conceptual

Un modelo conceptual captura los tipos más importantes de objetos en el contexto del sistema. Los objetos del dominio representan las "cosas" que existen o los eventos que suceden en el entorno en el que trabaja el sistema (15).

Provee un marco de referencia para la práctica, aunque la metodología XP no propone artefactos en este sentido, es flexible y puede contrastarse con algunos que simplifiquen y apoyen el proceso, al unísono esclarezcan los conceptos referentes al problema antes mencionado.

Los principales conceptos identificados en este dominio fueron:

- COLM: existe un Centro "Oscar Lucero Moya".
- DSD: existe un Departamento Secretaría Docente que pertenece al COLM.
- Secretario: existen secretarios que registran personas y subsistemas.
- Persona: pueden ser estudiantes o profesores.
- Profesor: persona encargada de impartir las asignaturas.
- Estudiante: persona encargada de recibir las asignaturas.
- Subsistema: hace referencia a las especialidades del sector de la construcción vinculadas al proceso docente del COLM. Un subsistema contiene cursos y diplomados.
- Curso: hace referencia a los cursos que pertenecen a un subsistema. El curso tiene modalidades y asignaturas.
- Diplomado: hace referencia a los diplomados que pertenecen a un subsistema. El diplomado está

compuestos por módulos.

- Modalidad: hace referencia a las diferentes modalidades que tiene un curso. Las modalidades pueden ser habilitación o perfeccionamiento.
- Habilitación: modalidad que tiene un curso para los estudiantes que no tienen conocimientos ni habilidades de la especialidad.
- Perfeccionamiento: modalidad que tiene un curso para los estudiantes que poseen conocimientos y habilidades de la especialidad o transita de categoría.
- Módulo: hace referencia a los módulos compuestos por un diplomado. El módulo tiene asignaturas.
- Asignatura: hace referencia a las asignaturas que tiene un curso o un módulo. Las asignaturas son impartidas por los profesores y recibidas por los estudiantes.

La Figura 2 muestra el modelo conceptual donde se refleja el funcionamiento del proceso de gestión de información académica de la UEB Centro de Preparación de Operarios "Oscar Lucero Moya". Se define el flujo de este proceso y la cantidad de instancias siguiendo la dirección de las relaciones.

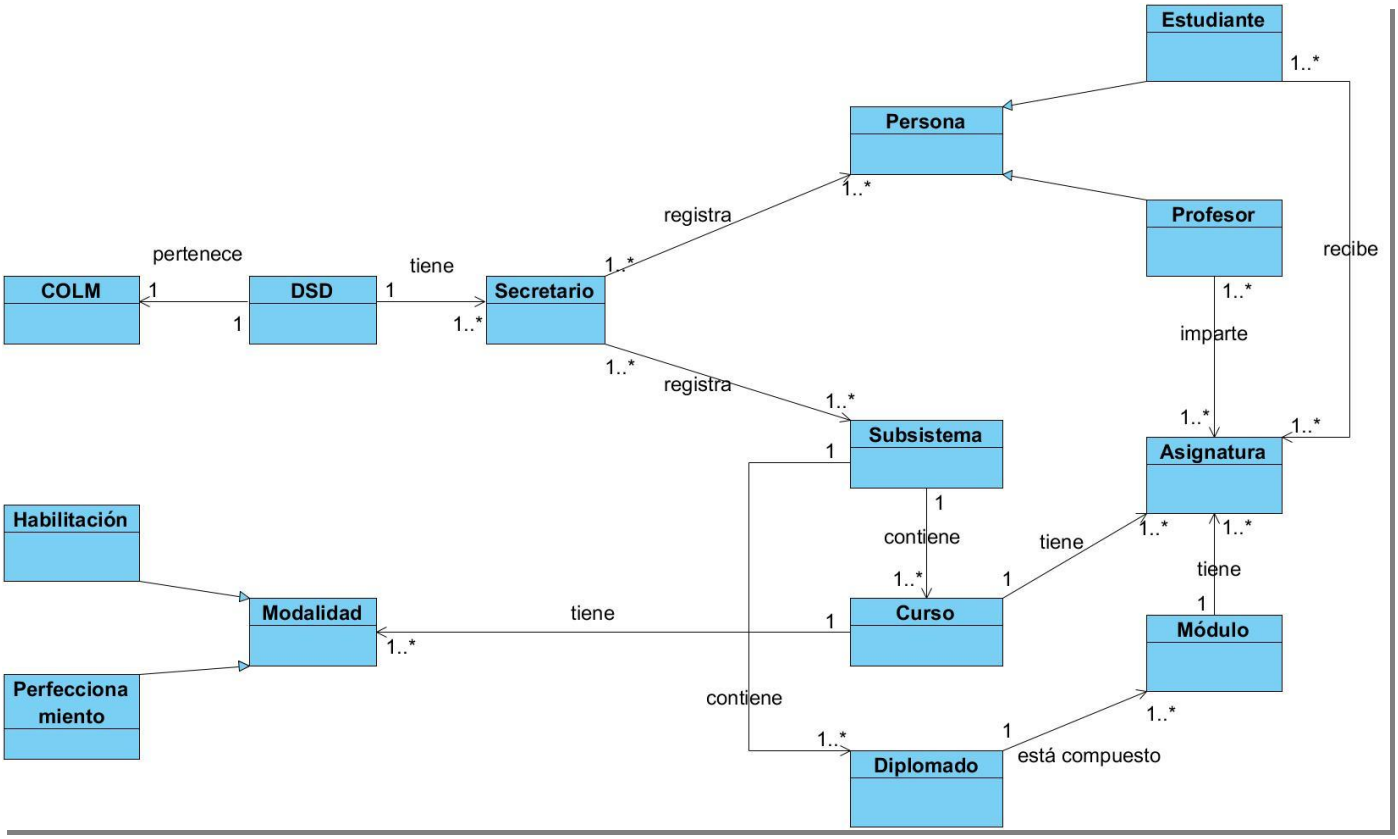


Figura 2. Modelo conceptual.

2.2 Propuesta del sistema

Teniendo en cuenta los requerimientos planteados y para cumplir los objetivos propuestos, se concibió como propuesta de solución la implementación de un sistema que contribuya a la mejora de la gestión de información académica en el Departamento Secretaría Docente de la UEB Centro de Preparación de Operarios "Oscar Lucero Moya". Dicho sistema es una aplicación web dinámica que utiliza como estructura de soporte el framework Symfony2 integrado con el servidor web Apache, con una autenticación por roles permisibles a realizar operaciones en dependencia del tipo de usuario autenticado, esto garantiza la seguridad de los datos manejados. Posibilitará visualizar informes, planillas, reportes y graficar estadísticas de la información académica esgrimidos en dicho departamento.

Una vez autenticado el usuario, el sistema le proporcionará permisos de acuerdo a su rol y por ende acceso a las funcionalidades del sistema correspondiente. El sistema cuenta con 5 roles, directivo,

secretario, profesor, estudiante y administrador del sistema.

En resumen, con esta propuesta de sistema se facilitará la centralización, manejo y control de la información del proceso gestión de información académica en el Departamento Secretaría Docente de la UEB Centro de Preparación de Operarios "Oscar Lucero Moya".

2.2.1 Personas relacionadas con el sistema

Una de las bases fundamentales que se deben tener en cuenta cuando se comienza el desarrollo de un sistema informático es la delimitación de la audiencia a la cual va dirigido el mismo, lo cual en algunas ocasiones, puede ser un personal relacionado con el sistema o alguien completamente ajeno a él. Ha de especificarse que esta audiencia, a su vez, puede ser dividida en grupos atendiendo a sus necesidades.

Dentro de la audiencia a interactuar con el sistema a desarrollar se incluyen aquellas personas que de una manera u otra interactúan con este, obteniendo un resultado de uno o varios procesos que se ejecutan en el mismo. También son considerados como personas relacionadas con el sistema, aquellas involucradas en dichos procesos, pero no obtienen ningún resultado de valor.

Tabla 2. Personas relacionadas con el sistema.

Personas relacionadas con el sistema	Justificación
Directivo	Dispone de posibilidades ilimitadas para ejecutar todas las funciones del sistema, además de ver información existente en el sistema.
Secretario	Dispone de posibilidades ilimitadas para ejecutar las funciones del sistema, además de ver información existente en el sistema.
Profesor	Dispone de posibilidades limitadas, solo evaluar a los estudiantes, además de ver información existente en el sistema.
Estudiante	Dispone de posibilidades limitadas, solo podrá ver información existente en el sistema.
Administrador	Es la persona facultada para la gestión del sistema con algunas limitantes. Se encarga de gestionar la administración de usuarios, roles y demás elementos del sistema.

2.3 Flujos de trabajo para el desarrollo del sistema

2.3.1 Lista de reserva del producto

La lista de reserva del producto en general son condiciones o capacidades necesarias por los usuarios para resolver un problema o alcanzar un objetivo. Se pueden clasificar en requisitos funcionales y no funcionales. A continuación se muestra la captura de requisitos realizada para desarrollar la solución propuesta.

2.3.2 Requisitos funcionales

Los requisitos funcionales son aquellos que describen qué debe hacer el sistema desde el punto de vista de las necesidades del usuario, son capacidades o condiciones que debe cumplir el sistema y que están fuertemente ligados a las opciones del programa. La Tabla 3 muestra algunos de los principales requisitos funcionales del sistema. El anexo (Anexo 2) describe el resto de los requisitos identificados.

Tabla 3. Principales requisitos funcionales del sistema.

Requisitos funcionales	
RF 1. Listar usuarios	RF 59. Eliminar curso
RF 2. Agregar usuario	RF 60. Listar diplomados
RF 3. Agregar dirección	RF 61. Agregar diplomado
RF 4. Visualizar detalles	RF 62. Agregar módulo al diplomado
RF 5. Modificar usuario	RF 63. Visualizar detalles del diplomado
RF 6. Eliminar usuario	RF 64. Modificar diplomado
RF 7. Autenticar usuario en el sistema "SGIA".	RF 65. Eliminar diplomado
RF 54. Listar cursos	RF 66. Listar subsistemas
RF 55. Agregar curso	RF 67. Agregar subsistema
RF 56. Agregar asignatura al curso	RF 68. Modificar subsistema
RF 57. Visualizar detalles del curso	RF 69. Eliminar subsistema
RF 58. Modificar curso	RF 71. Listar asignaturas

2.3.3 Requisitos no funcionales

Los requisitos no funcionales son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, rápido o confiable. En muchos casos los requerimientos no funcionales son fundamentales en el éxito del producto.

- **Requerimiento de hardware**

RnF 1. El servidor de base de datos debe tener como mínimo: un procesador Intel Pentium Dual-Core T2330 a 1,60 GHz, 2GB de memoria RAM y 1TB de disco duro.

RnF 2. El servidor para la instalación del sistema informático de gestión "SGIA" debe tener como mínimo: un procesador Intel Pentium Dual-Core T2330 a 1,60 GHz, 2GB de memoria RAM y 1TB de disco duro.

RnF 3. Las estaciones de trabajo cliente deben tener como mínimo: un procesador Intel Celeron 430 a 1.6 GHz y 1GB de memoria RAM.

- **Seguridad:**

RnF 4. La seguridad del sistema desarrollado debe estar basada en niveles de acceso sobre las funcionalidades y la información manejada por el sistema.

RnF 5. La seguridad se establecerá por perfiles, los cuales serán asignados a los usuarios que interactúen con el sistema, para garantizar de esta forma que la información almacenada solo sea modificada y/o visualizada por los usuarios autorizados.

RnF 6. El sistema debe responder a la política de acceso basada en roles, usuarios, permisos de los usuarios para las operaciones y permisos funcionales sobre las operaciones.

RnF 7. Se usarán mecanismos de encriptación de los datos que por cuestiones de seguridad no deben viajar al servidor en texto plano, como es el caso de las contraseñas.

RnF 8. Para prevenir la pérdida de datos el administrador del sistema realizará la salva diaria de la base de datos en el soporte correspondiente.

- **Usabilidad**

RnF 9. El sistema utilizará el idioma español para los mensajes y textos de la interfaz.

RnF 10. El sistema debe contar con una información organizada de manera que pueda ser utilizado de forma satisfactoria por la mayoría de los usuarios.

RnF 11. El sistema deberá brindar facilidad de uso por parte de los desarrolladores del software.

- **Confidencialidad**

RnF 12. Existencia de roles que establezcan el correcto acceso a la información según su rol.

RnF 13. Restringir la ejecución de acciones a usuarios sin credenciales que intenten acceder a las mismas y verificar que el usuario esté autenticado antes de realizar cualquier operación.

- **Software**

RnF 14. Los servidores deberán tener instalado los sistemas operativos Windows, Unix o Linux.

RnF 15. Las PC clientes deberán disponer de un navegador web, estos pueden ser Internet Explorer 7 o superior, Opera 9 superior, Google Chrome 1 o superior y Firefox 2 o superior.

- **Interfaz de usuario**

RnF 16. El sistema deberá poseer una interfaz gráfica uniforme, estructurado por módulos los cuales reúnen todas las funcionalidades de la aplicación, a las que se pueden acceder mediante un menú desplegable, estos módulos y funcionalidades se encuentren en un panel ubicado a la izquierda en cada vista del sistema.

RnF 17. Deberá contar con una adecuada combinación de colores claros como el gris y el azul, para lograr una armonía y un equilibrio haciendo que el diseño sea más efectivo. La tipografía usada será Calibri, pensada para su uso en pantalla, sobre todo en páginas web.

2.3.4 Fase de exploración

La metodología de desarrollo Programación Extrema (XP) comienza con su fase de exploración. Es la etapa del proceso de desarrollo de software que propone XP para comenzar la construcción de un producto. En esta fase, los clientes plantean a grandes rasgos las Historias de Usuario (HU), las cuales son de interés para la primera entrega del producto. Al mismo tiempo el desarrollador se familiariza con las herramientas, tecnologías y prácticas que utilizadas en el proyecto (14).

Una vez que los clientes entregan su propuesta al equipo de trabajo, comienza el análisis en grupo. Aspecto clave en el esclarecimiento de las dudas sobre los procesos a informatizar es: desde el inicio del proyecto, un miembro del equipo de trabajo del cliente se una al equipo de desarrolladores.

Historias de usuario (HU)

Las HU sustituyen a los documentos de especificación funcional, y a los "casos de uso", son como descripciones cortas de lo que el sistema debe realizar (52).

Para la confección de las HU se realizó entrevistas a los asesores técnico-docentes de la UEB Centro de

Preparación de Operarios "Oscar Lucero Moya". El cliente no es quién escribe las HU, si diseñó su contenido, asignó la prioridad de cada una de ellas para tener en cuenta la velocidad con la cual se desarrollaran las funcionalidades y dirigió la redacción de las mismas. El objetivo de las HU no se alteró, porque se logró la brevedad y claridad de las mismas.

El desarrollador por su parte revisó la prioridad, analizó la dependencia entre las HU y asignó el costo a cada una de ellas, el costo se traduce en las semanas para su desarrollo.

Se realizaron 24 HU. La Tabla 4 muestra la HU número 11 Gestionar curso. El resto de las HU identificadas se pueden ver en el anexo (Anexo 3).

Tabla 4. HU Gestionar curso.

Historia de Usuario	
Número: 11	**Nombre de la Historia de Usuario:** Gestionar curso.
Referencia:	
Usuario: administrador	**Iteración asignada:** 1
Prioridad en negocio: Alta	**Puntos estimados:** 1.0
Riesgo en desarrollo: Alta	**Puntos reales:** 1.0
Descripción: Para la gestión de un curso, el administrador una vez registrado en el sistema se desplazará hacia el Módulo Curso/Diplomado y dará clic sobre él, donde se desplegará una lista con los grupos Cursos y Diplomados, además de la opción Subsistemas. Al dar clic en el grupo Cursos, se desplegarán las opciones: Listado de cursos y Agregar curso. Luego si el administrador selecciona la opción: 1. Listado de cursos: el sistema muestra la vista Cursos, donde se listarán los cursos. Luego en esa misma vista si el administrador realiza la acción de: • Agregar curso: el sistema muestra una vista con los campos que deben ser llenados, luego de introducir los datos, se muestra un mensaje de información "Curso registrado" quedándose la vista activa para brindar la posibilidad de agregar otro. • Modificar curso: al dar clic en el botón Modificar correspondiente al curso deseado, el sistema muestra una vista que permite modificar los datos del cuso, al modificar los datos deseados, estos se salvan y el sistema muestra el mensaje de información "Curso modificado" y se muestra el listado actualizado de los cursos. • Agregar asignatura al curso: al dar clic en el botón Asignaturas correspondiente al curso deseado, el sistema muestra una vista con los campos de la asignatura que deben ser llenados.	

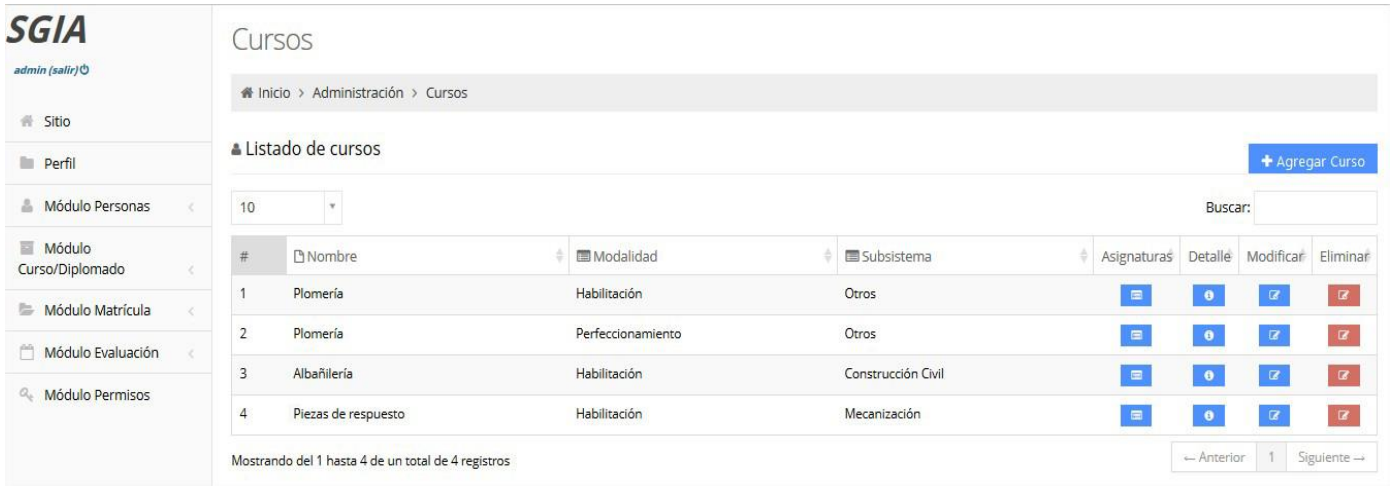
- Visualizar detalles del curso: al dar clic en el botón Detalle correspondiente al curso deseado, el sistema muestra una vista con los datos del curso.
- Eliminar curso: al dar clic en el botón Eliminar correspondiente al curso deseado, el sistema muestra el mensaje de confirmación "¿Seguro desea eliminar el curso para esta modalidad en este subsistema?". Se elige la opción aceptar y se muestra un mensaje de información "Eliminando".

2. Agregar curso: el sistema muestra una vista con los campos que deben ser llenados, luego de introducir los datos, se muestra un mensaje de información "Curso registrado" quedándose la vista activa para brindar la posibilidad de agregar otro.

Observaciones:

- Los cursos tienen una o varias asignaturas asociadas.
- Las modalidades pueden ser: Habilitación o Perfeccionamiento.
- Un curso no puede ser eliminado si tiene datos asociados a una edición.

Prototipo de interfaz de usuario:



2.3.5 Fase de planificación y entrega

En la fase de planificación se priorizan las HU y se acuerda el alcance de cada una de las entregas o release. Los desarrolladores estiman cuanto esfuerzo requiere cada HU y a partir de allí se define el plan de iteraciones. El cliente decide las HU que se seleccionan para cada iteración. Las pruebas funcionales creadas por el cliente se ejecutan al final de cada iteración (14).

Prioridad de las Historias de Usuario

La asignación de prioridad a las HU definidas por el cliente, decidió el orden en el cual se implementarán las mismas, siempre y cuando estén de acuerdo usuario y desarrollador. Las HU con menor número de prioridad son las más importantes; quedan conformadas de la siguiente manera:

Tabla 5. Prioridades por HU.

Historia de usuario	Prioridad Asignada
Gestionar usuario	1
Autenticar usuario	1
Gestionar rol de usuario	1
Gestionar profesor	1
Gestionar profesor inactivo	1
Gestionar estudiante	1
Gestionar estudiante baja	1
Gestionar dirección	1
Gestionar graduación	1
Gestionar centro de procedencia	1
Gestionar curso	1
Gestionar diplomado	1
Gestionar subsistema	1
Gestionar asignatura	2
Gestionar módulo	2
Gestionar asignatura del modulo	2
Gestionar edición	2
Gestionar curso de la edición	2
Gestionar grupo de clase	2
Gestionar turno de clase	2
Gestionar evaluación	2
Desarrollar listados y reportes	3

Visualizar listados y reportes	3
Graficar estadísticas de reportes	3

Estimación de esfuerzos por Historias de Usuario

Las estimaciones del esfuerzo para implementar las HU permitieron tener una medida bastante real de la velocidad de progreso del proyecto, pues brindaron una guía razonable a la cual ajustarse. La Tabla 6 resume la estimación del esfuerzo realizada por parte del desarrollador.

Tabla 6. Estimación de esfuerzo por HU.

Nro.	Historias de usuario	Puntos de estimación (semanas)
1	Gestionar usuario	1.0
2	Autenticar usuario	1.0
3	Gestionar rol de usuario	0.4
4	Gestionar profesor	1.0
5	Gestionar profesor inactivo	1.0
6	Gestionar estudiante	1.0
7	Gestionar estudiante baja	1.0
8	Gestionar dirección	0.2
9	Gestionar graduación	0.2
10	Gestionar centro de procedencia	0.4
11	Gestionar curso	0.5
12	Gestionar diplomado	0.5
13	Gestionar subsistema	0.3
14	Gestionar asignatura	0.5
15	Gestionar módulo	0.4
16	Gestionar asignatura del módulo	0.3
17	Gestionar edición	0.5
18	Gestionar curso de la edición	0.5
19	Gestionar grupo de clase	0.5
20	Gestionar turno de clase	0.5

21	Gestionar evaluación	0.2
22	Desarrollar listados y reportes	1.0
23	Visualizar listados y reportes	1.0
24	Graficar estadísticas de reportes	1.0

Después del análisis, el desarrollo total estimado contó con 14.9 semanas, los valores de cada una de las HU estuvieron comprendidos entre 0.2 y 1.0 semanas en dependencia de la complejidad de cada una de ellas.

Plan de iteraciones

Luego de identificar y redactar cada una de las HU, establecer la prioridad y la estimación del esfuerzo necesario para realizarlas, se debe conformar el plan de iteraciones. Las HU seleccionadas para cada iteración son desarrolladas y probadas de acuerdo al orden preestablecido. El desarrollo fue dividido en tres iteraciones, para las cuales se desarrollaron partes funcionales de la aplicación.

A continuación se describen cada una de las iteraciones mencionadas y la justificación de su selección:

- Iteración 1: para esta iteración se desarrollan las HU de mayor importancia (prioridad alta) para el cliente, necesarias para comenzar la implementación de la iteración siguiente. Estas HU son desde la 1 a la 13 correspondientes a gestionar usuario, autenticar usuario, gestionar rol de usuario, gestionar profesor, gestionar profesor inactivo, gestionar estudiante, gestionar estudiante baja, gestionar dirección, gestionar graduación, gestionar centro de procedencia, gestionar curso, gestionar diplomado y gestionar subsistema. Las HU seleccionadas para esta iteración permiten la gestión de los elementos básicos del sistema, los cuales no todos tienen dependencias durante su desarrollo y son de gran importancia para las futuras iteraciones. En esta iteración se asegura la seguridad del sistema mediante la autenticación de los usuarios, dependiendo de los permisos que posea y se realizan las pruebas de aceptación. Al finalizar la iteración, se realiza una entrega funcional solicitada por el cliente en la cual se puede realizar la gestión de los elementos independientes del sistema.

- Iteración 2: en esta iteración se desarrollan las HU de importancia media para el cliente. Estas HU son desde la 14 a la 21 correspondientes a gestionar asignatura, gestionar módulo, gestionar asignatura del módulo, gestionar edición, gestionar curso de la edición, gestionar grupo de clase, gestionar turno de clase y gestionar evaluación. En esta iteración se realizan las pruebas de aceptación y la entrega

funcional al cliente, la cual permite la gestión de las ediciones con los subsistemas, cursos y modalidades asociadas a ellas, además del proceso de matrícula y baja de los estudiantes y los turnos de clases que serán impartidos por los profesores.

- Iteración 3: en esta iteración se desarrollan las HU de importancia baja para el cliente. Estas HU son desde la 22 a la 24 correspondientes a desarrollar listados y reportes, visualizar listados y reportes y graficar estadísticas de reportes. En la iteración, al igual que en las anteriores se realizan las pruebas de aceptación. Al finalizar la iteración se realiza la entrega final del producto.

Para tener un tiempo estimado de ejecución de cada iteración, se tomó como medida que cada semana cuenta con 6 días (lunes, martes, miércoles, jueves, viernes y sábado) en los que se trabajaban 8 horas sin distracciones. En la Tabla 7 se muestra el plan de iteraciones, este incorpora el tiempo estimado para cada una de las iteraciones y las HU que se van a desarrollar. La primera iteración es más extensa por incluir las HU de mayor complejidad.

Plan de duración de las iteraciones

Como parte del ciclo de vida de desarrollo de un producto de software guiado por la metodología XP, se creó el plan de duración de cada una de las iteraciones, así como el orden en que serán implementadas las HU en cada una de las mismas. Este plan ayudará a tener una noción aproximada del tiempo que durará el desarrollo del sistema en su totalidad. La Tabla 7 muestra las 3 iteraciones analizadas previamente, con las HU que incluyen y su duración:

Tabla 7. Plan de iteraciones.

Iteración	Historias de usuario	Puntos de estimación (semanas)
1	Gestionar usuario	8.5
	Autenticar usuario	
	Gestionar rol de usuario	
	Gestionar profesor	
	Gestionar profesor inactivo	
	Gestionar estudiante	
	Gestionar estudiante baja	

	Gestionar dirección	
	Gestionar graduación	
	Gestionar centro de procedencia	
	Gestionar curso	
	Gestionar diplomado	
	Gestionar subsistema	
2	Gestionar asignatura	3.4
	Gestionar módulo	
	Gestionar asignatura del módulo	
	Gestionar edición	
	Gestionar curso de la edición	
	Gestionar grupo de clase	
	Gestionar turno de clase	
	Gestionar evaluación	
3	Desarrollar listados y reportes	3.0
	Visualizar listados y reportes	
	Graficar estadísticas de reportes	

Plan de entregas

El plan o cronograma de entregas establece que HU son agrupadas para conformar una entrega, y el orden de las mismas. El cronograma de entregas se realiza sobre la base de las estimaciones de tiempos de desarrollo realizadas por los desarrolladores (14).

A partir del plan de iteraciones analizado en el acápite anterior, y en correspondencia con el mismo se realizó el plan de entregas, que se muestra en la Tabla 8. Con dos versiones funcionales y una última entrega de iteraciones del producto en la fecha el 10 de junio, para dar paso a la fase de implantación.

Tabla 8. Plan de entregas.

Historias de usuario	Final 1ra Iteración 23/03/2015	Final 2da Iteración 04/05/2015	Final 3ra Iteración 10/06/15

Gestionar usuario Autenticar usuario Gestionar rol de usuario Gestionar profesor Gestionar profesor inactivo Gestionar estudiante Gestionar estudiante baja Gestionar dirección Gestionar graduación Gestionar centro de procedencia Gestionar curso Gestionar diplomado Gestionar subsistema	V1.0	Finalizado	Finalizado
Gestionar asignatura Gestionar módulo Gestionar asignatura del módulo Gestionar edición Gestionar curso de la edición Gestionar grupo de clase Gestionar turno de clase Gestionar evaluación	No empezado	V1.1	Finalizado
Desarrollar listados y reportes Visualizar listados y reportes Graficar estadísticas de reportes	No empezado	No empezado	V1.2

2.4 Diseño del sistema

XP establece prácticas especializadas que inciden directamente en la realización del diseño para lograr un sistema robusto y reutilizable tratando de mantener su simplicidad, es decir, crear un diseño evolutivo que se perfecciona, permite hacer entregas pequeñas y frecuentes de valor para el cliente. Al cumplimentar la actividad de diseño, XP no específica ninguna técnica de modelado, puede utilizar indistintamente sencillos esquemas en una pizarra, diagramas de clases utilizando UML o tarjetas CRC (Clase, Responsabilidad y Colaboración) siempre que sean útiles, tributen a la comprensión y no requieran mucho tiempo en su creación (14).

2.4.1 Tarjetas CRC

Las tarjetas CRC (Clase - Responsabilidad - Colaboración) son confeccionadas durante la etapa de diseño de la metodología de desarrollo XP. La misma constituye un modelo simple que tiene como objetivo desarrollar una representación organizada de las clases. Cada tarjeta representa una clase, de cada una se describe sus responsabilidades y las clases colaboradoras que se relacionan con la misma. Las responsabilidades son los atributos y las operaciones relevantes para la clase. Los colaboradores son aquellas clases requeridas para que una clase reciba la información necesaria para completar una responsabilidad (13).

Se realizó un total de 21 Tarjetas CRC. La Tabla 9 muestra la tarjeta CRC CursoModalidad. El resto de las tarjetas se pueden ver en el anexo (Anexo 4).

Tabla 9. Tarjeta CRC CursoModalidad

Tarjeta CRC	
Clase: CursoModalidad	
Responsabilidad:	**Colaboración:**
Permite crear un curso. Permite crear un diplomado. Permite obtener un listado de los cursos. Permite obtener un listado de los diplomados. Permite modificar los datos de un curso. Permite modificar los datos de un diplomado. Permite buscar un curso por modalidad y subsistema. Permite eliminar un curso.	CursoModalidadType CursoModalidad Curso Subsistema Modalidad

2.4.2 Arquitectura

Una arquitectura de software define la estructura del sistema, la cual adquiere gran importancia debido a que las representaciones de arquitecturas de software permiten la comunicación entre todas las partes interesadas en el desarrollo de cómputo; constituye un modelo comprensible de cómo está estructurado el sistema y cómo trabajan juntos sus componentes (60).

2.4.3 Patrones

Los patrones y estilos son comportamientos que deben existir en el sistema, ayudan a describir qué es lo que debe hacer, es decir, describen el uso del sistema y cómo este interactúa con los usuarios.

Entre sus características se encuentran: describir el problema de forma sencilla, describir el contexto en el que ocurre, puntualizar los pasos a seguir, hacer énfasis en los puntos fuertes y débiles de la solución, referir otros patrones asociados, entre otras. En la presente investigación se abordan los patrones arquitectónicos y los patrones de diseño que son utilizados para conformar el diseño de la aplicación propuesta (53).

Patrón de arquitectura

Un patrón arquitectónico constituye una descripción de un problema particular y recurrente de diseño, que aparece en contextos de diseño específico, y presenta un esquema genérico demostrado con éxito para su solución. Expresa el esquema de organización estructural fundamental para sistemas de software. Provee un conjunto de subsistemas predefinidos, especifica sus responsabilidades e incluye reglas y pautas para la organización de las relaciones entre ellos. Es una plantilla para arquitecturas de software concretas, que especifica las propiedades estructurales de una aplicación (61).

El framework seleccionado para el desarrollo del sistema a implementar, está totalmente basado en la arquitectura Modelo Vista Controlador (MVC). Esta arquitectura permite dividir la aplicación en tres grandes capas (39):

- **Vista:** Es lo que utilizan los usuarios para interactuar con la aplicación (los gestores de plantillas pertenecen a esta capa).
- **Modelado:** Es la capa del modelo define la lógica de negocio (la base de datos pertenece a esta capa).
- **Controlador:** Es un bloque de código que realiza llamadas al modelo para obtener los datos y se los pasa a la vista para que los muestre al usuario.

En Symfony, el acceso y la modificación de los datos que se almacenan en la base de datos, se realiza mediante objetos. La implementación que realiza Symfony de la arquitectura incluye clases como:

- **Controller:** es la clase del controlador y se encarga de decodificar la petición y transferirla a la acción

correspondiente.

- **Request:** guarda todos los elementos que integran la petición (parámetros, cookies, cabeceras.)
- **Response:** posee las cabeceras de la respuesta y los contenidos. El contenido de este objeto se convierte en la respuesta HTML que se remite al usuario.

El cliente envía una señal llamada request o petición, ésta es interceptada por el controlador quien realiza las validaciones necesarias, procesamiento de dichos datos y lógica de negocio asociadas a esa petición del cliente. El controlador envía datos al modelado, por ejemplo para ser guardados en una base de datos y/o los obtiene dependiendo de la solicitud del usuario para finalmente enviarlos a la vista a fin de ser mostrador nuevamente al cliente a través de un response o respuesta. Ver Figura 3.

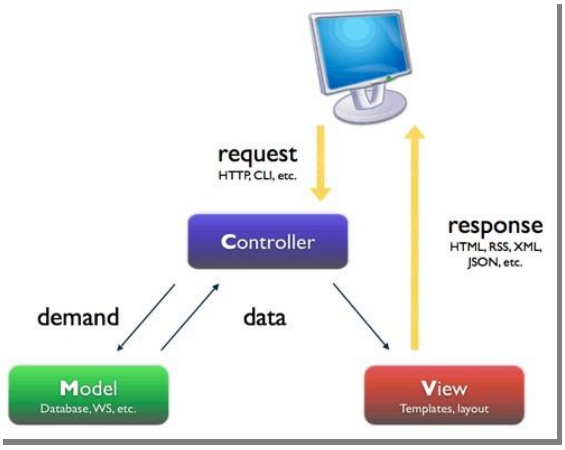


Figura 3. Funcionamiento del patrón arquitectónico MVC en Symfony.

Patrones de diseño

Un patrón de diseño ayuda a mejorar la calidad de un software en términos de que el mismo sea reusable, fácil de darle mantenimiento, extensibles y además de reducir el tiempo de desarrollo. Son usualmente independientes del lenguaje de programación a utilizar. Los patrones de diseño son empleados generalmente durante el proceso de creación de un framework o sea estos internamente encierran muchos patrones de diseño presentes en la solución propuesta a través del framework Symfony2 (53).

Para mejorar la calidad de la propuesta de solución en términos de que el mismo sea reusable, fácil de mantener, extensibles, además de reducir el tiempo de desarrollo, se utilizaron diferentes patrones de diseño. A continuación se explica brevemente cuales fueron utilizados.

Patrones de asignación de responsabilidades o GRASP

Son una ayuda de aprendizaje que permiten al desarrollador entender lo esencial del diseño de objetos y aplicar el razonamiento del mismo de una forma metódica, racional y explicable. Este enfoque se entiende en el uso de los principios del diseño basado en patrones para la asignación de responsabilidades a las clases y objetos de una aplicación (53).

Siguiendo el patrón arquitectónico MVC utilizado por Symfony2, a continuación se describe la utilización de los patrones GRASP:

Experto en información

Este patrón plantea que la asignación de responsabilidades será para la clase que contiene la información necesaria para cumplir la responsabilidad. Permite conservar el encapsulamiento ya que los objetos se valen de su propia información para hacer lo que se les pide, lo que provee un bajo nivel de acoplamiento. Promueve clases sencillas y cohesivas que son más fáciles de mantener y comprender. Este patrón es utilizado en la capa de abstracción del modelo de datos a través del ORM Doctrine. Las clases generadas poseen un grupo de funcionalidades que facilitan el acceso y la manipulación de los datos de las entidades persistentes en la base de datos (53). A continuación se presenta un ejemplo con la clase entidad "Profesor.php", ubicada en la carpeta: sgia\src\AppBundle\Entity\ e implementada en la propuesta:

```php
<?php

namespace AppBundle\Entity;

use Doctrine\ORM\Mapping as ORM;
use Symfony\Component\Validator\Constraints as Assert;
use Doctrine\Common\Collections\ArrayCollection;

/** Profesor ...6 lines */
class Profesor
{
    /**
     * @var integer
     *
     * @ORM\Column(name="id_profesor", type="integer", nullable=false)
     * @ORM\Id
     * @ORM\GeneratedValue(strategy="IDENTITY")
     */
```

Figura 4. Patrón Experto en información. Declaración de la clase Profesor.

Creador

Este patrón resuelve el problema de asignar responsabilidades relacionadas con la creación de objetos. Es utilizado en los controladores, en ellos se encuentran las acciones definidas para el sistema (53). En la

implementación de las acciones se crean los objetos de las clases que presentan las entidades evidenciando que las clases controladoras son creadoras de dichas entidades. En la aplicación se evidencia en la implementación de la clase "ProfesorController.php" ubicada en la carpeta: sgia\src\AppBundle\Controller\Admin\.

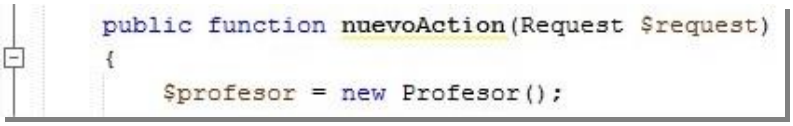


Figura 5. Patrón Creador. Función donde se crea un objeto de la clase Profesor.

Bajo acoplamiento

Bajo acoplamiento significa que una clase no depende de muchas clases. Las clases controladoras no dependen de otras clases para realizar sus funcionalidades. Los modelos solo dependen de la clase controladora que lo usa, así se pueden realizar cambios en cada clase de forma independiente (53). En la aplicación se evidencia en la implementación de la clase "ProfesorController.php" ubicada en la carpeta: sgia\src\AppBundle\Controller\Admin\ que hereda de la clase "Controller.php" ubicada en la carpeta sgia\src\AppBundle\Controller\. La clase controladora "ProfesorController.php" realiza las funciones para la gestión de un profesor, como son: crear profesor, listar profesores, mostrar profesores inactivos, desactivar profesor, activar profesor, ver detalles del profesor, modificar profesor y eliminar profesor.

Alta cohesión

El diseño de clases se realizó de forma tal que cada clase tenga responsabilidades moderadas en un área funcional. Cada una de las clases posee solamente las responsabilidades imprescindibles. Se agruparon funciones similares en clases individuales para poder lograr una alta cohesión. La cohesión no puede considerarse individualmente sin considerar también el bajo acoplamiento y el patrón experto en información (53). En la aplicación se evidencia en la implementación de las clases entidades, ubicadas en la carpeta: sgia\src\AppBundle\Controller\Entity.

Controlador

Consiste en un objeto que no pertenece a la interfaz de usuario, responsable de recibir o manejar un evento del sistema, define el método para la operación del sistema (53). En Symfony se evidencia en las clases que forman la capa Controlador del patrón arquitectónico MVC, ubicadas en la carpeta: sgia\src\AppBundle\Controller\Admin\. Estas clases controladoras definen los métodos para las

operaciones del sistema.

Patrones GoF

Los patrones GoF (Gang of Four) son una serie de posibles soluciones a problemas que suelen ser comunes en el desarrollo del software, se dividen en tres grandes categorías: creacionales, estructurales y de comportamiento (54).

A continuación se describe la utilización de algunos de los patrones GoF:

Patrón creacional Fábrica (Factory)

Involucra algún tipo de factoría o fábrica de objetos. Los objetos de fabricación tienen la función de crear instancias de otras clases. Además tienen la responsabilidad y el conocimiento necesario para encapsular la forma en que se crean determinados tipos de objetos en una aplicación. Existen distintos tipos de patrones de fabricación, tales como: simple factory, factory method y abstract factory (55). Este patrón se empleó para crear objetos de tipo formulario. En la aplicación se evidencia en la implementación de la clase "ProfesorController.php", ubicada en la carpeta: \sgia\src\AppBundle\Controller\Admin\, la cual se relaciona con el formulario "ProfesorType.php", ubicado en la carpeta: sgia\src\AppBundle\Form\. El método createForm de la clase base se encarga de crear el formulario, pasándole el "tipo" de formulario, así como los datos iniciales.

```
public function nuevoAction(Request $request)
{
    $profesor = new Profesor();

    $formulario = $this->createForm(new ProfesorType(), $profesor);
    $formulario->handleRequest($request);
```

Figura 6. Patrón Fábrica.

Patrón estructural Adaptador (Adapter)

Su propósito es convertir la interfaz de una clase en otra interfaz que el cliente espera. El Adaptador permite a las clases trabajar juntas, lo que de otra manera no podría hacerlo debido a sus interfaces incompatibles (53). Se utilizó en la configuración al crear un formulario. En la aplicación se evidencia en los formularios ubicados en la carpeta: sgia\src\AppBundle\Form\.

```
public function setDefaultOptions(OptionsResolverInterface $resolver)
{
    $resolver->setDefaults(array(
        'data_class' => 'AppBundle\Entity\Profesor',
        'csrf_message' => 'Token de seguridad invalido',
        'intention' => 'authenticate'
    ));
}
```

Figura 7. Patrón Adaptador.

Patrón estructural Decorador (Decorator)

Este patrón añade funcionalidad a una clase de una forma dinámica y transparente. Decorator hace que el código sea reutilizable (53). En la aplicación se evidencia en la plantilla "backend.html.twig" ubicada en la carpeta: sgia\app\Resources\views. Este archivo almacena el código HTML que es común en mucha de las páginas de la aplicación para no tener que repetirlo en otra página. El contenido de la misma se integra a las plantillas heredadas por ejemplo en las plantillas "index.html.twig" ubicada en las carpetas de: sgia\app\Resources\views\.

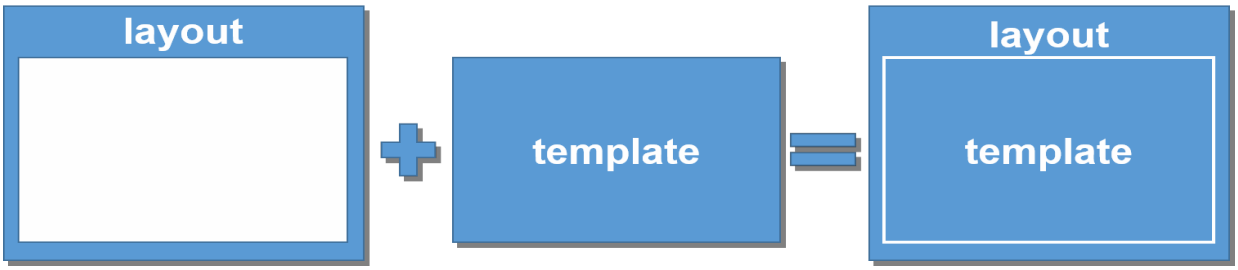


Figura 8. Patrón Decorador en Symfony.

Patrón estructural Fachada (Facade)

Este patrón viene motivado por la necesidad de estructurar un entorno de programación y reducir su complejidad con la división en subsistemas, minimizando las comunicaciones y dependencias entre estos. Su principal ventaja consiste en que para modificar las clases de los subsistemas, sólo hay que realizar cambios en la interfaz/fachada, y los clientes pueden permanecer ajenos a ello, o sea los clientes no necesitan conocer las clases que hay tras de la interfaz (53). Se aplicó el patrón fachada para proporcionar una interfaz simple para un subsistema complejo, o estructurar varios subsistemas en capas, ya que las fachadas serían el punto de entrada a cada nivel (Ver Figura 9). En la aplicación se evidencia en la implementación de la clase "TurnoController.php", ubicada en la carpeta:

\sgia\src\AppBundle\Controller\Admin\.

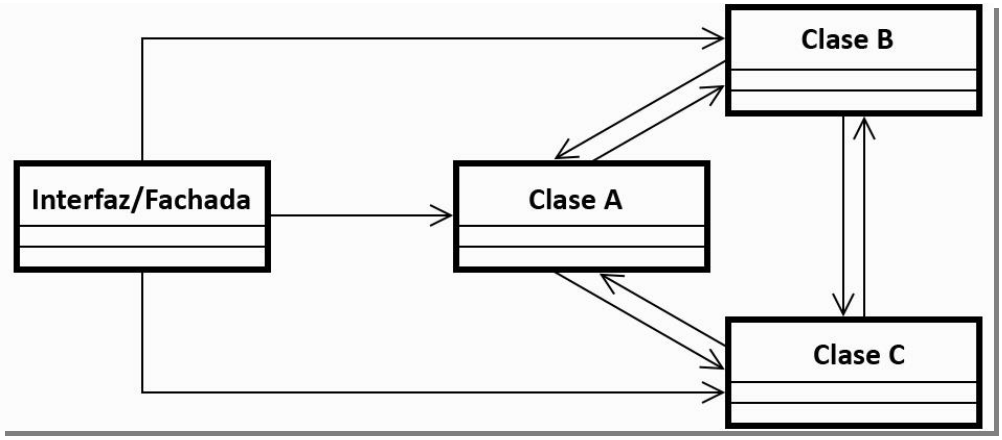


Figura 9. Patrón Fachada en Symfony.

2.4.4 Diseño de la base de datos

En cualquier sistema en el que se gestione información la base de datos desempeña un papel fundamental. El diseño de la misma es un proceso esencial para el desarrollo de sistemas de este tipo, pues permite el almacenamiento de la información de forma coherente y organizada, para evitar que dicha información se pierda. Es por esto que se realizó el diseño de la base de datos, para lograr la persistencia de los datos de los usuarios, así como la información de los recursos que se gestionan por los mismos.

Debido al tamaño de la base de datos manejada, se muestra el módulo persona como fracción elegida del modelo físico de la base de datos obtenida para el sistema "SGIA". Este módulo se seleccionó pues su implementación fue de gran importancia dada por el cliente durante la iteración 1. Ver Figura 9. El modelo físico de la base de datos cuenta con un total de 23 tablas, se puede ver en el anexo (Anexo 5). El anexo (Anexo 6) describe cada una de las entidades que conforma el modelo de datos.

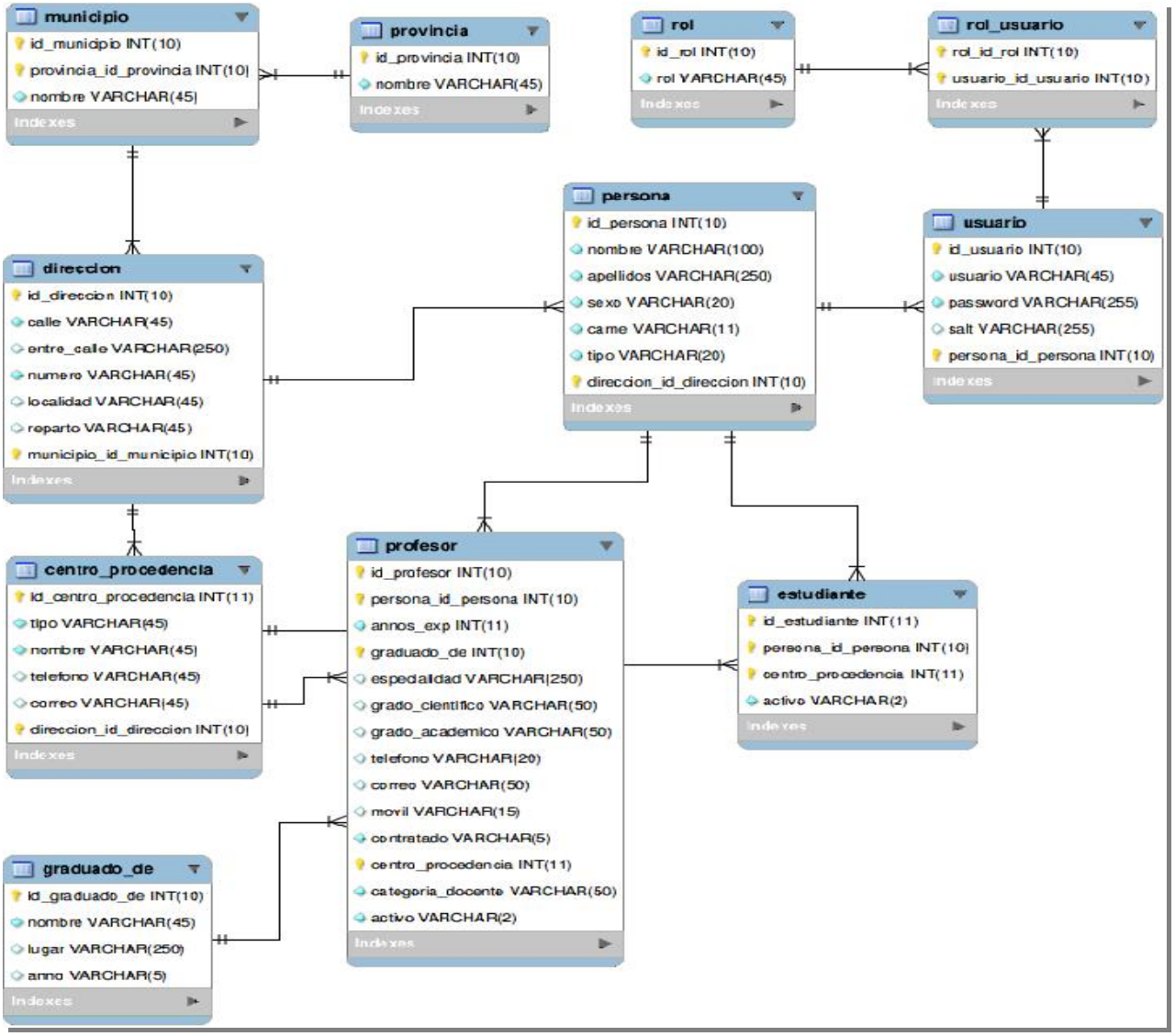


Figura 10. Módulo persona seleccionado de la base de datos obtenida para el sistema "SGIA".

2.4.5 Diagrama de despliegue

El diagrama de despliegue es un grafo de nodos unidos por conexiones de comunicación que muestra las relaciones físicas entre los componentes de hardware y forman la topología sobre la que se ejecuta el sistema y la distribución de sus partes. Este refleja tanto la distribución del sistema con los nodos físicos (componentes) como la correspondencia de los componentes con los nodos (53).

La figura muestra el entorno de despliegue de la aplicación que refleja los componentes y sus relaciones: uno como servidor para la base de datos, otro para que funcione como servidor web y otro que representa en general todos los clientes que puedan conectarse al sistema a través del navegador web. Los nodos que representan el servidor de base de datos y el servidor web deben ser capaces de brindar una serie de prestaciones de hardware para cumplir con los requerimientos.

El entorno de despliegue de la aplicación queda estructurado de la siguiente manera:

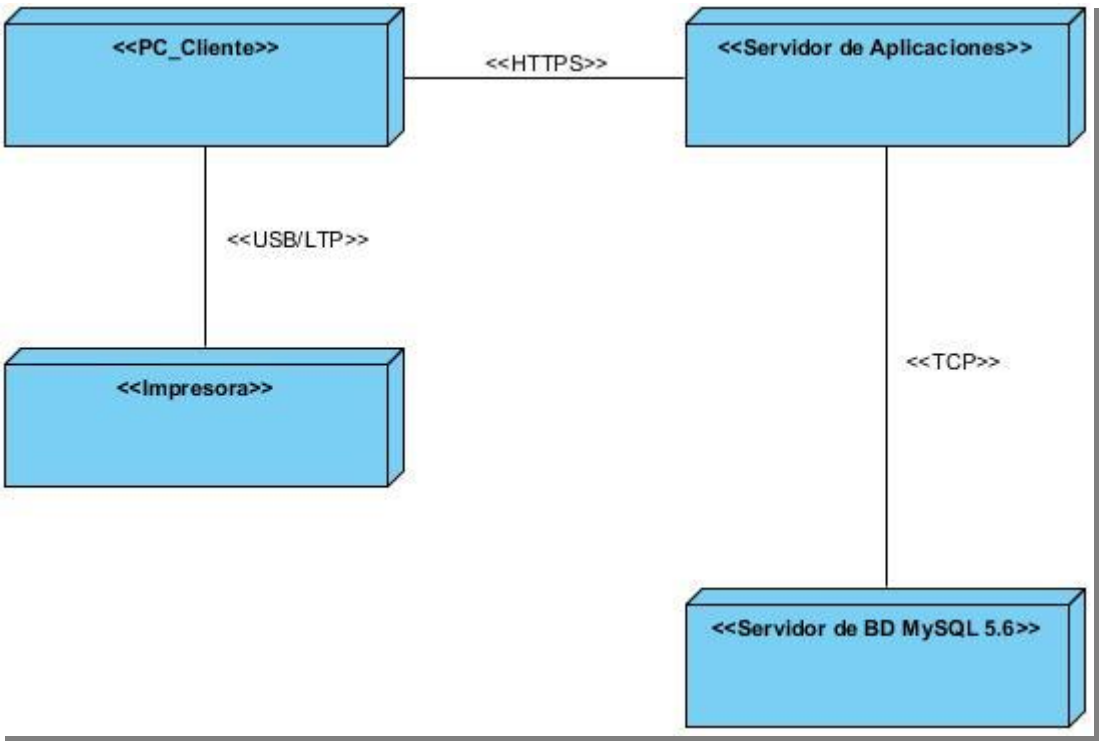


Figura 11. Diagrama de despliegue.

Conclusiones del capítulo

Después de la ejecución de las fases de exploración, planificación y la mitad de la fase entrega planteadas por la metodología de desarrollo de software XP, el autor arribó a las siguientes conclusiones parciales:

Después de aplicar el método científico entrevista, se realizó el levantamiento de la lista de reserva del producto, con 126 requisitos clasificándolos en requisitos funcionales y requisitos no funcionales. La lista de reserva del producto permitió: identificar las principales características y funcionalidades que debe tener el sistema. En la fase de exploración, se definieron las funcionalidades del sistema mediante el desarrollo de 24 HU. En la fase de planificación y entrega, se estableció la prioridad a cada HU, la estimación de esfuerzos por HU. Se definió el plan de iteraciones con tres iteraciones realizadas, obteniéndose el plan de duración de las iteraciones y el plan de entregas con tres entregas funcionales. Como parte del diseño del sistema se realizó la descripción de las principales clases con un total de 21 tarjetas CRC para después garantizar una mejor calidad del código en la fase de implementación, se aplicó el patrón de arquitectura MVC y los patrones de diseño GRASP y GoF. Además se definió el modelo físico de la base de datos con un total de 23 tablas y se muestra la vista de despliegue en un ambiente real indicando los elementos fundamentales y el protocolo de comunicación que se utiliza entre ellos.

Capítulo 3: Implementación y pruebas de la aplicación

Introducción

Tras investigar los elementos referentes al diseño de la aplicación a desarrollar, una de las siguientes fases es la implementación, guiada por las tareas en las que se disecciona una UH para buscar una mayor fiabilidad del producto. La fase de pruebas le permite al desarrollador, realizar ensayos constantemente del producto después de finalizar cada iteración, intentando lograr la ausencia de errores.

3.1 Implementación del sistema

Las tareas llevadas a cabo en la fase de implementación, fueron escritas utilizando un lenguaje técnico y no necesariamente entendible por el cliente. Esta labor fue puesta en práctica para detallar mejor las HU, facilita el entendimiento en el proceso de implementación. Cada HU puede contener una o más tareas y explica de forma general las acciones que se realizan en la misma.

3.1.1 Tareas de ingeniería

Durante el transcurso de las iteraciones fue efectuada la implementación de las HU diseñadas por el cliente y descritas por el desarrollador en la etapa de exploración. Como parte del plan, se diseccionaron las HU en tareas de la ingeniería, las cuales son asignadas a los programadores para ser implementadas durante la iteración correspondiente. Una HU puede tener asociada varias tareas de ingeniería para poder darle respuesta a las necesidades de su cliente.

La Tabla 10 muestra las tareas de ingeniería que estarán presente en la mayoría de las HU.

Tabla 10. Tareas genéricas.

Tareas de ingeniería
➢ Diseñar base de datos.
➢ Generar las clases modelo para la conexión a la base de datos.
➢ Implementar los métodos en las clases controladoras.
➢ Sincronizar los datos con la base de datos.
➢ Definir sistema de enrutamiento.
➢ Definir componentes para la interfaz.

Las tareas de ingeniería de cada una de ellas se muestran en el anexo (Anexo 8).

La Tabla 11 muestra la relación de algunas de las tareas de ingeniería por HU de la iteración 1, escogidas por ser las de mayor importancia dada por el cliente, pues muestran la mayor parte de los procesos asociados a las actividades de la UEB Centro de Preparación de Operarios "Oscar Lucero Moya", elementos indispensables para la realización del sistema. Cada tarea de desarrollo corresponde a un período de medio día a tres días de desarrollo.

Tabla 11. Relación de las tareas de ingeniería por HU de la iteración 1 y de gran importancia dada por el cliente.

Historias de Usuario	Puntos reales	Puntos estimados	No.	Tareas de Ingeniería
Gestionar usuario	1.0	1.0	1	Listar usuarios
			2	Agregar usuario
			3	Agregar dirección
			4	Visualizar detalles
			5	Modificar usuario
			6	Eliminar usuario
Autenticar usuario	1.0	1.0	7	Crear plantilla para formulario de autenticar
			8	Validar los campos del formulario
			9	Verificar los datos insertados
			10	Definir los privilegios del usuario
Gestionar curso	1.0	1.0	57	Listar cursos
			58	Agregar curso
			59	Agregar asignatura al curso
			60	Visualizar detalles del curso
			61	Modificar curso
			62	Eliminar curso
Gestionar diplomado	1.0	1.0	63	Listar diplomados
			64	Agregar diplomado
			65	Agregar módulo
			66	Visualizar detalles
			67	Modificar diplomado
			68	Eliminar diplomado

Las demás tareas de ingeniería por HU de cada iteración se pueden ver en el anexo (Anexo 7).

Se realizaron 114 tareas de ingeniería. En las tablas, de la 12 a la 15 se describen algunas de las tareas de ingeniería de la HU 11 Gestionar curso, perteneciente a la iteración 1.

Tabla 12. Tarea de ingeniería 57 de la iteración 1: Listar cursos.

Tarea de Ingeniería	
No. de tarea: 57	**No. de HU:** 11
Nombre de la tare a: Listar cursos.	
Tipo de tarea: Desarrollo.	**Puntos estimados: 0.15**
Fecha inicio: 07/03/15	**Fecha fin: 09/03/15**
Programador responsable: Félix Daniel Aguilera Pérez.	
Descripción: Se listan los cursos existentes en la base de datos.	

Tabla 13. Tarea de ingeniería 58 de la iteración 1: Agregar curso.

Tarea de Ingeniería	
No. de tarea: 58	**No. de HU:** 11
Nombre de la tare a: Agregar curso.	
Tipo de tarea: Desarrollo.	**Puntos estimados: 0.1**
Fecha inicio: 09/03/15	**Fecha fin: 09/03/15**
Programador responsable: Félix Daniel Aguilera Pérez.	
Descripción: Se agrega un curso a la base de datos.	

Tabla 14. Tarea de ingeniería 61 de la iteración 1: Modificar curso.

Tarea de Ingeniería	
No. de tarea: 61	**No. de HU:** 11
Nombre de la tare a: Modificar curso.	
Tipo de tarea: Desarrollo.	**Puntos estimados: 0.05**
Fecha inicio: 11/03/15	**Fecha fin: 11/03/15**
Programador responsable: Félix Daniel Aguilera Pérez.	

Descripción: Se modifica un curso existente en la base de datos.

Tabla 15. Tarea de ingeniería 62 de la iteración 1: Eliminar curso.

Tarea de Ingeniería	
No. de tarea: 62	**No. de HU:** 11
Nombre de la tare a: Eliminar curso.	
Tipo de tarea: Desarrollo.	**Puntos estimados: 0.05**
Fecha inicio: 12/03/15	**Fecha fin: 12/03/15**
Programador responsable: Félix Daniel Aguilera Pérez.	
Descripción: Se elimina un curso existente en la base de datos.	

En la primera iteración se realizó un total de 72 tareas de ingeniería, en la segunda 39 y en la última 3. En el anexo (Anexo 8), se puede observar el resto de las tareas hechas por iteración y en correspondencia con las HU.

3.1.2 Estándares de codificación

Los estándares de codificación son pautas 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 (56).

Seguir un determinado estilo de codificación permite a los programadores revisar, mantener y actualizar el código de una manera sencilla y ordenada, evitando que incurran en errores y malas prácticas que dificultan la comprensión de las líneas de código. Se muestran los estándares definidos para la implementación del sistema informático "SGIA".

Estructura (57)**:**

Añade un solo espacio después de cada delimitador coma.

Añade un solo espacio alrededor de los operadores (==, &&,...).

Añade una coma después de cada elemento del arreglo en un arreglo multilínea, incluso después del último.

Añade una línea en blanco antes de las declaraciones return, a menos que el valor devuelto solo sea

dentro de un grupo de declaraciones (tal como una declaración if).

Usa llaves para indicar la estructura del cuerpo de control, independientemente del número de declaraciones que contenga.

Define una clase por archivo.

Declara las propiedades de clase antes que los métodos.

Declara métodos públicos, luego los protegidos y finalmente los privados.

Utiliza paréntesis para instanciar clases sin importar el número de argumentos que tenga el constructor.

Convenciones de nomenclatura (57):

Utiliza mayúsculas intercaladas —sin guiones bajos— en nombres de variable, función, método o argumentos.

Usa guiones bajos para nombres de opción y nombres de parámetro.

Utiliza espacios de nombres para todas las clases.

Prefija las clases abstractas con Abstract.

Sufija las interfaces con Interface.

Sufija las características con Trait.

Sufija las excepciones con Exception.

Utiliza caracteres alfanuméricos y guiones bajos para los nombres de archivo.

Se muestra una parte de un código que contiene la mayoría de las funciones descritas anteriormente:

```
<?php

/*
 * Código de ejemplo.
 */

namespace Acme;

/**
 * Demostración de estándares de la codificación.
 */
class FooBar
{
    const SOME_CONST = 42;

    private $fooBar;

    /**
     * @param string $dummy Some argument description
     */
    public function __construct($dummy)
    {
        $this->fooBar = $this->transformText($dummy);
    }
    /**
     * @param string $dummy Some argument description
```

```
     * @param array $options
     *
     * @return string|null Transformed input
     */
    private function transformText($dummy, $options = array())
    {
        $mergedOptions = array_merge(
            $options,
            array(
                'some_default' => 'values',
                'another_default' => 'more values',
            )
        );
        if (true === $dummy) {
            return;
        }
        if ('string' === $dummy) {
            if ('values' === $mergedOptions['some_default']) {
                $dummy = substr($dummy, 0, 5);
            } else {
                $dummy = ucwords($dummy);
            }
        } else {
            throw new \RuntimeException(sprintf('Unrecognized dummy option "%s"', $dummy));
        }
        return $dummy;
    }
}
```

3.2 Pruebas de software

Las pruebas son un conjunto de actividades que se pueden planificar por adelantado y llevar a cabo sistemáticamente, por esta razón se deben definir en el proceso de ingeniería del software. Estas contribuyen a elevar la calidad de los productos desarrollados y a la seguridad de los programadores a la hora de introducir modificaciones (58).

La metodología XP propone el uso de pruebas de aceptación, destinadas a evaluar si al final de una iteración se obtuvo la funcionalidad requerida, además comprueban que dicha funcionalidad fuera la esperada por el cliente (14).

3.2.1 Pruebas de aceptación

Las pruebas de aceptación, también llamadas pruebas funcionales son supervisadas por el cliente basándose en los requerimientos tomados de las HU, su propósito es garantizar el cumplimiento de los requisitos pautados además de asegurar su correcto funcionamiento. Son consideradas como "pruebas de caja negra" y generalmente los clientes participan en estas pruebas, de quedar satisfechos con las demostraciones entonces la aplicación es entregada.

Estas pruebas se elaboraron a lo largo de cada iteración, en paralelo con el desarrollo del sistema, adaptándose a los cambios anteriores. Fueron realizadas para probar que el sistema recibiera un

conjunto de entradas y produjera salidas o respuestas (Ver Figura 11), haciendo mínimo enfoque en la estructura lógica interna del software comprobando que esté listo y pueda ser usado por los usuarios finales (59). El proceso de realización de estas pruebas culminó con la firma de conformidad del Acta de Aceptación por parte del cliente, se puede ver en el anexo (Anexo 10).

Figura 12. Funcionamiento del método de pruebas de caja negra.

3.2.2 Diseño de casos de prueba

Los casos de prueba son evidencias de pruebas funcionales que se le realizan al sistema para comprobar su funcionamiento y diseñados a partir de las HU de la propuesta de solución. Una HU puede tener tantos casos de prueba como sean necesarios para evaluar su funcionamiento y no se puede considerar terminada hasta tanto no pase todas las pruebas.

Como criterio de aprobación de cada iteración se tomó que el 100% de los casos de prueba fueran exitosos para pasar de iteración. El objetivo de estas pruebas en cada ciclo de iteración del desarrollo no fue tener un conjunto de casos de prueba escritos que cubran el 100% del código, sino haberle realizado pruebas al sistema desde el punto de vista del usuario. El cliente especificó uno o diversos escenarios para comprobar que una HU fue implementada correctamente.

Para la realización de cada una de las pruebas se siguió una serie de pasos, los cuales se muestran a continuación:

- Identificar todas las acciones en la HU.
- Para cada acción escribir al menos una prueba.
- Para algunos datos, reemplazar las entradas que hacen que la acción ocurra y llenar en la casilla resultados esperados los resultados obtenidos.
- Para otros datos, reemplazar las entradas que hacen que la acción falle, y registrar los resultados.

A continuación se muestran las iteraciones de pruebas elaboradas para el sistema informático "SGIA":
Iteración 1: la presente iteración de pruebas tuvo como objetivo comprobar la correcta implementación de varias HU y detectar posibles no conformidades, la relación de HU involucradas en el proceso fue la siguiente: gestionar usuario, autenticar usuario, gestionar rol de usuario, gestionar profesor, gestionar profesor inactivo, gestionar estudiante, gestionar estudiante baja, gestionar dirección, gestionar graduación, gestionar centro de procedencia, gestionar curso, gestionar diplomado y gestionar subsistema. Se le realizaron 69 casos de prueba a las HU de la iteración 1. Se muestran algunos de los casos de prueba realizados a las HU de dicha iteración, las demás se pueden ver en el anexo (Anexo 9).

Tabla 16. HU2_P7: Autenticar_usuario_en_el_sistema

Caso de prueba de aceptación: 7	
Código: HU2_P7	**HU:** 2_Autenticar_usuario.
Nombre: Autenticar_usuario_en_el_sistema	
Responsable: Félix Daniel Aguilera Pérez.	
Descripción: Probar que se pueda autenticar un usuario en el sistema.	
Condiciones de Ejecución: El usuario debe poseer algún rol para que pueda entrar al sistema.	
Entrada/Pasos de Ejecución: El usuario accede al sistema, donde inserta su nombre de usuario y contraseña. **Pasos:** Acceder a la página de Inicio del sistema y seleccionar la opción Autenticarse.	
Resultado esperado: Si el nombre de usuario y contraseña es válido. Se le muestra una página al usuario según sus privilegios, en la cual se le brindan las funcionalidades correspondientes a su rol. Si el nombre de usuario y contraseña no es válido. El sistema informa al usuario que su usuario y/o contraseña no son válidos, y se le da la posibilidad de intentarlo nuevamente.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 17. HU11_P56: Agregar_curso.

Caso de prueba de aceptación: 56	
Código: HU11_P56.	**HU:** 11_Gestionar curso.
Nombre: Agregar_curso.	

Responsable: Félix Daniel Aguilera Pérez.
Descripción: Prueba de funcionalidad para la inserción de un curso en el sistema.
Condiciones de Ejecución: El usuario debe estar autenticado y tener permisos para realizar esta operación.
Entrada/Pasos de Ejecución: Se selecciona la opción Agregar curso, mostrándose un formulario con los campos correspondientes a este. **Pasos:** Se tienen dos opciones para la inserción de un curso: Opción 1: Acceder al módulo Curso/Diplomado. Acceder al grupo Cursos. Seleccionar la opción Listado de cursos y luego dar clic en el botón Agregar curso. Opción 2: Acceder al módulo Curso/Diplomado. Acceder al grupo Cursos y seleccionar la opción Agregar curso.
Resultado esperado: Si el usuario ha introducido todos los datos del curso. El curso es registrado sin generar algún error. Si el usuario omite seleccionar la modalidad del curso. La aplicación lanzará la alerta "Por favor, seleccione la modalidad del curso". Si el usuario omite seleccionar el subsistema del curso. La aplicación lanzará la alerta "Por favor, escriba el nombre del subsistema del curso". Si el usuario inserta algún dato existente al agregar el curso. La aplicación lanzará la alerta "Ya existe un curso con estas características".
Evaluación de la prueba: Prueba satisfactoria.

Tabla 18. HU11_P59: Modificar_curso.

Caso de prueba de aceptación: 59	
Código: HU11_P59.	**HU:** 11_Gestionar curso.
Nombre: Modificar_curso.	
Responsable: Félix Daniel Aguilera Pérez.	
Descripción: Prueba de funcionalidad para modificar un curso en el sistema.	
Condiciones de Ejecución: El usuario debe estar autenticado y tener permisos para realizar esta operación.	

Entrada/Pasos de Ejecución: Se da clic en el botón Modificar de un curso deseado, mostrándose un formulario con los campos correspondientes, debe permitirse modificar cualquiera de los datos pertenecientes al curso deseado. **Pasos:** Acceder al módulo Curso/Diplomado. Acceder al grupo Cursos. Acceder a la opción Listado de cursos y dar clic en el botón Modificar correspondiente al curso deseado.
Resultado esperado: Si el usuario modifica los datos del curso. El curso es modificado sin generar algún error. Si el usuario inserta algún dato existente al modificar el curso. La aplicación lanzará la alerta "Ya existe un curso con estas características".
Evaluación de la prueba: Prueba satisfactoria.

Iteración 2: la presente iteración de pruebas tuvo como objetivo comprobar la correcta implementación de varias HU para detectar posibles no conformidades, la relación de HU involucradas en el proceso fue la siguiente: gestionar asignatura, gestionar módulo, gestionar asignatura del módulo, gestionar edición, gestionar curso en edición, gestionar grupo de clase, gestionar turno de clase, gestionar turno impartido y gestionar evaluación. Se le realizaron 39 casos de prueba a las HU de la iteración 2, se pueden ver en el anexo (Anexo 9).

Iteración 3: la presente iteración de pruebas tuvo como objetivo comprobar la correcta implementación de varias HU para detectar posibles no conformidades, la relación de HU involucradas en el proceso fue la siguiente: desarrollo de los reportes, visualizar informes y graficar estadísticas. Se le realizaron 3 casos de prueba a las HU de la iteración 3, se pueden ver en el anexo (Anexo 9).

Iteración 4: el objetivo de esta iteración fue realizar todas las pruebas hecha por cada iteración para comprobar la correcta implementación de las HU. Después de realizadas las pruebas, se hizo un levantamiento de las no conformidades detectadas por el cliente.

3.2.3 Ejecución de los casos de prueba de aceptación

El proceso de pruebas del software se realizó a través de las iteraciones descritas anteriormente donde a medida que se procedió con una nueva iteración se erradicaron los defectos encontrados en la anterior, garantizando que al final del proceso el producto quede libre de la mayor cantidad de errores posibles y listo para entregárselo al cliente.

Durante la ejecución de los casos de prueba de aceptación algunas no arrojaron los resultados esperados. Se aplicaron un total de 111 casos de prueba que se distribuyeron en 4 iteraciones. La Tabla 19 muestra el resumen de los resultados de las No conformidades (NC) obtenidas, distinguiendo en ellas cuáles proceden, cuáles no proceden y las resueltas, resaltando que las NC encontradas no constituyeron un riesgo de desarrollo alto para el funcionamiento del sistema.

Tabla 19. Resumen de casos de prueba ejecutados y NC detectadas por iteración.

Iteraciones	Casos de prueba ejecutados	Total de NC	#NC No Proceden	#NC Resueltas
1	69	5	0	5
2	39	5	0	5
3	3	4	0	4
4	111	0	0	14

Iteración 1: se realizaron un total de 69 casos de prueba detectándose un total de 5 NC. A continuación se listan los casos de prueba de aceptación que arrojaron alguna NC durante la ejecución de la primera iteración de pruebas:

HU2_P8: Autenticar_con_datos_válidos.

HU4_P13: Listar_profesores.

HU4_P14: Agregar_profesor.

Iteración 2: se realizaron un total de 39 casos de prueba detectándose un total de 5 NC. A continuación se listan los casos de prueba de aceptación que arrojaron alguna NC durante la ejecución de la segunda iteración de pruebas:

HU14_P3: Agregar_asignatura

HU16_P15: Listar_ediciones

HU17_P21: Listar_cursos_de_la_edición

HU19_P39: Agregar_turno_de_clase

Iteración 3: se realizaron un total de 3 casos de prueba detectándose un total de 4 NC. A continuación se muestra el caso de prueba de aceptación que arrojó alguna NC durante la ejecución de la tercera iteración de pruebas:

HU22_P2: Visualizar_listados_y_reportes

Iteración 4: se realizaron todos los casos de prueba, no detectándose alguna NC.

La siguiente figura muestra la cantidad de NC detectadas en los casos de prueba ejecutados por iteración.

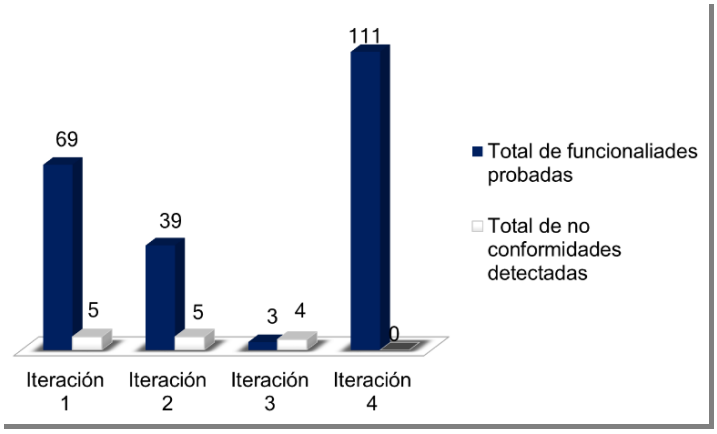


Figura 13. Total de no NC detectadas por iteración.

La plantilla de NC constituyó un registro de los defectos y fallos encontrados en el transcurso de las pruebas realizadas por iteración, su objetivo fue verificar en un futuro que estos errores fueron erradicados en las iteraciones realizadas anteriormente.

La investigación se dividió en tres iteraciones, la Tabla 20 muestra las NC encontradas en cada una de ellas:

Tabla 20. Resumen de NC.

Iteraciones	Descripción
1	1. No se especifica el mensaje de información que muestra al usuario los errores en el formulario al autenticarse.
	2. Errores de falta de ortografía en los mensajes de información mostrados al usuario en la pantalla de autenticación.
	3. Errores de falta de ortografía a la hora de mostrar los mensajes de información que se muestran cuando se quiere eliminar un profesor.
	4. A la hora de agregar un profesor no se queda la vista activa brindando la posibilidad de agregar otro profesor.
	5. Al dar clic en el botón Listar profesores, este no lista y redirecciona a la misma página que estaba antes.

2	1. La funcionalidad activar o desactivar una edición no está implementada y muestra la interfaz para con el formulario para modificar una edición.
	2. No estaban implementados los tooltips de los botones del listado de las Ediciones
	3. Al desactivar un determinado curso perteneciente a una edición se mostraba el listado de los curso de esa edición en el listado de las ediciones.
	4. Al agregar un turno de clase no se definieron las funcionalidades para un turno de clase en un determinado curso con su modalidad y subsistema.
	5. Errores de falta de ortografía al mostrar los mensajes de información cuando el usuario dejaba algún campo vacío.
3	1. La funcionalidad para ver los estudiantes de un determinado grupo de clase no estaba implementado.
	2. Errores de falta de ortografía a la hora de mostrar los listados de los cursos de la edición actual.
	3. No muestra el reporte cumplimiento de matrícula de los centros de procedencia.
	4. No muestra el reporte cumplimiento de matrícula de graduados por cursos de subsistemas.

3.2.4 Resultados

Para que el proceso de pruebas tenga éxito se requirió del análisis final de los resultados arrojados en cada iteración realizada, es decir, la evaluación del producto que se está probando de acuerdo a todos los defectos y fallos del sistema encontrados a lo largo del proceso.

Las pruebas de aceptación se realizaron en cuatro iteraciones y se utilizaron los 111 casos de prueba diseñados, para verificar que las funcionalidades implementadas fueron las acordadas en las HU y respondían correctamente a sus necesidades. A continuación se muestran los resultados obtenidos en las pruebas realizadas en las 4 iteraciones:

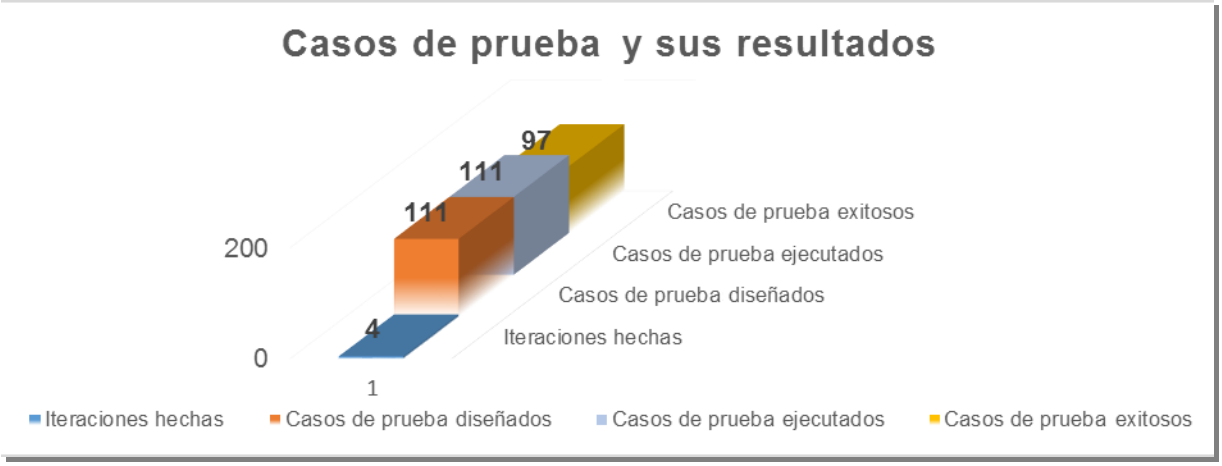


Figura 14. Resumen de los casos de prueba y sus resultados.

De los 111 casos de prueba diseñados, 97 resultaron exitosos, existiendo 14 que lanzaron una NC. Por tal motivo, una vez resueltos los defectos detectados se pasó a una cuarta iteración de pruebas, donde se volvieron a realizar todos los casos de prueba diseñados, para verificar que fueron resueltas las NC de la primera, segunda y tercera iteración y que además no habían sufrido cambios los casos de prueba exitosos de las mismas. Al concluir la cuarta iteración, todos los casos de prueba arrojaron resultados satisfactorios.

Conclusiones del capítulo

En este capítulo se definió el estándar de codificación a tener en cuenta para implementar el sistema "SGIA". El cumplimiento de las tareas de ingeniería dio solución a las historias de usuario. Se efectuaron las pruebas de aceptación logrando resultados satisfactorios. Se puede concluir, teniendo en cuenta los resultados arrojados por las pruebas realizadas, que el sistema "SGIA" se encuentra listo para su funcionamiento.

Conclusiones

- Se realizó un estudio de sistemas de gestión de información académica para confeccionar los fundamentos teóricos y se concluyó que no era posible utilizar algún sistema de gestión estudiado.

- Se estudiaron y seleccionaron metodologías, tecnologías y herramientas más idóneas para el desarrollo del sistema y se realizó el diseño del mismo haciendo uso de la metodología de desarrollo de software XP, patrón de arquitectura y patrones de diseño obteniéndose como resultado un diseño sólido y flexible para la codificación de la aplicación.

- Se desarrolló el Sistema de Gestión de Información Académica (SGIA) que contribuirá a la mejora de la gestión de información académica en el Departamento Secretaría Docente de la UEB Centro de Preparación de Operarios “Oscar Lucero Moya”.

- Se realizaron pruebas de aceptación propuesta por la metodología de desarrollo de software XP al Sistema de Gestión de Información Académica (SGIA), demostrando que está listo para su funcionamiento.

Recomendaciones

Tomando como base la investigación realizada y la experiencia acumulada durante el desarrollo del presente trabajo de diploma se recomienda:

- Elaborar un manual de usuario para uso de los clientes, con el objetivo de lograr un mayor entendimiento del sistema así como, su funcionamiento.
- Implementar la funcionalidad que permita exportar reportes que visualiza el Sistema de Gestión de Información Académica (SGIA) a otros formatos.

Referencias

1. Dante. Gestión de la información en las organizaciones: principios, conceptos y aplicaciones. Santiago de Chile : s.n., 1998.

2. Aplicaciones de Gestión de información en las organizaciones. El profesional de la información y su dominio de las técnicas y herramientas de la Gestión. La habana : Tesis de doctorado. Cuba, Departamento de Bibliotecología y Ciencia de la Información, Universidad de La Habana. 2000.

3. Urdaneta. ¿Qué es la gestión de Información? 1990.

4. BSI International. bsi. [En línea] 2013. [Citado el: 20 de Enero de 2015.] http://www.bsigroup.com.mx/es-mx.

5. Miño, Noel. Análisis y diseño del Módulo Estudiante para el sistema de gestión Académica- Akademos. [En línea] 2007. http://bibliodoc.uci.cu/TD/TD_0325_07.pdf..

6. Mireydis, Yurisleis. Análisis y Diseño de los Procesos de Gestión de Personal para Akademos v2.0. [En línea] 2013. [Citado el: 20 de Enero de 2015.] http://bibliodoc.uci.cu/TD/TD_1934_09.pdf.

7. Sistema académico de la Universidad Nacional Autónoma de Nicaragua. [En línea] 2014. https://www.sigacad.unanleon.edu.ni/.

8. SIGA. Sistema de Información de Gestión Académica. [En línea] 2013. https://siga.usm.cl//pag/portada_que_es.html

9. Vega Tabares, Iván; Aroche Dominguez, Taondris. Estrategia de selección de metodologías de software ágil o robusta. Tesis de pregrado. [En línea]

10. Brito Acuña, Karenny. Selección de metodolgías de desarrollo para aplicaciones web en la facultad de informática de la Universidad de Cienfuegos. 2009.

11. Letelier, Patricio; CANOS, José H; PENADÉS, María C. Metodolgías ágiles en el desarrollo de software. [En línea] 2013. [Citado el: 20 de Enero de 2015.] http: //noqualityinside.com.ar/nqi/nqifiles/XP_Agil.pdf.

12. Santiago, María Lourdes. Proceso de Desarrollo Unificado (RUP). [En línea] 2009. [Citado el: 20 de Enero de 2015.] http://www.utvm.edu.mx/OrganoInformativo/orgJul07/RUP.htm.

13. Beck, K. Extreme Programming Explained. 1999.

14. Joskowicz, José. Reglas y prácticas en Extreme Programming. [En línea] Universidad de Vigo, 2008. [Citado el: 14 de Febrero de 2015.] http: //iie.fing.edu.uy/~josej/docs/XP%20-%20Jose%20Joskowicz.pdf.

15. Universidad Luterana Salvadoreña (ULS). Reglas y Prácticas en eXtreme Programming. [En línea] 10 de Marzo de 2008. [Citado el: 16 de Febrero de 2015.] http://www.uls.edu.sv/index.php?option=com_phocadownload&view=category&download=90:re .

16. Villafuerte, Victor. Extreme Programming. eXtreme Programing explained . [En línea] 2009. [Citado el: 15 de Febrero de 2015.] http://extremeprogramming.host56.com/ARTICULO5.php .

17. Scuh, P. Recovery, redemption, and extreme programming. IEEE Software. 2001.

18. Larman, Craig. UML y Patrones. Introducción al análisis y diseño orientado a objetos. Mexico : PRENTICE HAL. ISBN: 970-1 7-0261-1.

19. Baisley, B;. Unified Modeling Language Infraestructure. s.l. : Pearson Editorial, 2006.

20. Pressman, R. Ingeniería del software: un enfoque práctico (5 ed.). 2005.

21. Rational. Rational. [En línea] 3 de Marzo de 2011. [Citado el: 20 de Febrero de 2015.] http://www.rational.com.ar/herramientas/roseenterprise.html.

22. Mitecnologico. Mitecnológico. [En línea] 13 de Mayo de 2011. [Citado el: 16 de Febrero de 2015.] http://www.mitecnologico.com/Main/DefinicionDeLenguajeDeProgramacion.

23. Peña, Mayra. Aplicaciones WEB. Scribd. [En línea] 2012. [Citado el: 16 de Febrero de 2015.] http://es.scribd.com/doc/136052164/APLICACIONES-WEB-pdf.

24. González, Benjamín. Guerra de lenguajes de Programación: Ruby vs Python vs PHP. [En línea] 2012. [Citado el: 16 de Febrero de 2015.] http://codigolinea.com/2012/09/13/guerra-de-lenguajes-de-programacion-ruby-vs-python-vs-php/ .

25. Gauchat, Juan Diego. El gran libro de HTML5, CSS3 y Javascript. [En línea] 2012. [Citado el: 17 de Febrero de 2015.] http://www.minkbooks.com . ISBN 978-84-267-1782-5.

26. Zeldman, Jeffrey. Diseño con estándares Web. s.l. : Ediciones ANAYA MULTIMEDIA, 2004.

27. Bbrandedaugh, J. Aplicaciones JavaScript . Madrid, España : O'Reilly & Associates, 2000. ISBN 84-415-1070-9.

28. Gallego Vázquez, Juan Diego. Desarrollo Web con PHP y MySQL. s.l. : Ediciones ANAYA MULTIMEDIA, 2006.

29. Mendoza, Ing. Iván. Trabajo de tesis para la obtención del título de Máster en Ingeniería de Software. Definición de un Framework para aplicaciones Web. [En línea] La Plata: Universidad Nacional de La Plata, 2010. [Citado el: 20 de Febrero de 2015.] http://sedici.unlp.edu.ar/bitstream/handle/10915/4192/Documento_completo.pdf?sequence=1 .

30. Gutiérrez, Javier J. ¿Qué es un framework web? . [En línea] 2012. [Citado el: 20 de Febrero de 2015.] http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf .

31. González, Benjamín. Comparación y Rendimiento de Frameworks PHP. [En línea] 2008. [Citado el: 20 de Febrero de 2015.] http://codigolinea.com/2008/06/04/compararcion-y-rendimiento-de-frameworks-php/ .

32. Anton, Oscar. Comparativa de frameworks. [En línea] 2011. [Citado el: 22 de Febrero de 2015.] http://www.moleculartS.com/blog/2011/06/01/comparativa-de-frameworks-en-php .

33. Micaela. Frameworks PHP recomendados, guía para principiantes. elWebmaster.com. [En línea] 2009. [Citado el: 22 de Febrero de 2015.] ttp://www.elwebmaster.com/articulos/frameworks-php-recomendados-guia-para- principiantes .

34. Powered by WordPress and zBench. PHPLand y otros pensamientos. [En línea] 2014. [Citado el: 22 de Febrero de 2015.] http://www.ricardclau.com/2011/06/elegir-un-framework-de-php-para-un-nuevo-proyecto/.

35. Terms of Yii Documentation. The Definitive Guide to Yii 2.0. [En línea] 2014 (c) Yii Software LLC, 21 de Marzo de 2014. [Citado el: 22 de Febrero de 2015.] http://www.yiiframework.com/doc-2.0/guide-index.html.

36. SensioLabs. http://symfony.com/doc/current/book/index.html. [En línea] [Citado el: 24 de Febrero de 2015.] http://symfony.com/pdf/Symfony_book_2.7.pdf?v=4.

37. Eguíliz, Javier. Desarrollo web ágil com Symfony2. [En línea] Javier. EasyBook, 2011. [Citado el: 24 de Febrero de 2015.] http://symfony.com/doc/2.0/book/index.html .

38. Potencier, Fabien. Libros WEB. Symfony 1.4, la guía definitiva. [En línea] 2015. [Citado el: 25 de Febrero de 2015.] http://librosweb.es/symfony_1_4/ .

39. Comunidad PHP. PHP. Yaml - Manual. [En línea] 2013. [Citado el: 25 de Febrero de 2015.] http://www.php.net/manual/en/book.yaml.php.

40. Pacheco, Nacho (tran.). Contenedor de servicios — Manual de Symfony2 en Español. [En línea] 2011. [Citado el: 26 de Febrero de 2015.] http://gitnacho.github.io/symfony-docs-es/book/service_container.html.

41. Akbari, Alí. PHP ORMs: Doctrine vs. Propel. [En línea] 2012. [Citado el: 26 de Febrero de 2015.] http://stackoverflow.com/questions/2062473/php-orms-doctrine-vs-propel.

42. Symfony Comunidad. Capítulo 8. La base de datos y Doctrine. [En línea] 2011. [Citado el: 4 de Marzo de 2015.] http://librosweb.es/symfony_2_x/capitulo_8.html .

43. Comunidad Symfony. Contenedor de servicios - Manual de Symfony2 en Español. [En línea] 2012. [Citado el: 4 de Marzo de 2015.] http://gitnacho.github.io/symf ony-docs-es/book/service_container.html .

44. Elcodigok. ElCodigok. [En línea] 14 de Mayo de 2011. [Citado el: 5 de Marzo de 2015.] http://www.elcodigok.com.ar/2010/09/caracteristicas-de-un-excelente-entorno-de-desarrollo-integrado..

45. BITSBETA. BitsBeta. [En línea] 5 de Diciembre de 2012. [Citado el: 5 de Marzo de 2015.] http://bitsbeta.com/netbeans-ide-grafica-programacion.

46. López Michelone, Manuel. PhpStorm. [En línea] 2013. [Citado el: 8 de Marzo de 2015.] http://www.unocero.com/2013/10/25/se-libera-phpstorm-8/.

47. Gamma, Erich; Helm, Richard; Johnson, Ralph; Vlissides, John. GoF Design Patterns with examples using Java and UML 2. [En línea] 2008. [Citado el: 8 de Marzo de 2015.] http://www.etnassoft.com/biblioteca/gof-design-patterns/. ISBN 0-201-63361-2.

48. Conocimiento virtual. Code Igniter Spanish User Guide. [En línea] Colombia, 2012. [Citado el: 8 de Marzo de 2015.] http://www.conocimientovirtual.edu.co.

49. Microsoft. Microsoft. [En línea] 2012. [Citado el: 9 de Marzo de 2015.] http://www.microsoft.com/spain/windowsserver2003/technologies/webapp/iis.mspx. .

50. Gallego, Jose Antonio. Desarrollo Web con PHP y My SQL. . ANAYA : Multimedia, 2006.

51. MySQL. MySql. [En línea] 2012. [Citado el: 10 de Marzo de 2015.] http://www.mysql.com/why-mysql.

52. Pérez, J. Guía Comparativa de Metodologías Ágiles. Segovia: Universidad de Valladolid : s.n., 2012.

53. Larman, Craig. UML y Patrones. Introducción al análisis y diseño orientado a objetos. Mexico : s.n. PRENTICE HAL. ISBN: 970-1 7-0261-1.

54. Guerrero, Carlos A.; Suarz, Johanna M.; Gutiérrez, Luz E. Patrones de Diseño GoF (The Gang of Four) en el contexto de Procesos de Desarrollo de Aplicaciones Orientadas a la Web. 2013. Vol. 24, no. 3, p. 103–114. DOI 10.4067/S0718-07642013000300012.

55. Gamma, E.; Helm, R.; Johnson, R.; VLI, J. Patrones de Diseño: elementos de software orientados a objetos reutilizables. s.l. : (Acebal, Trad.) Addison Wesley, 2002.

56. Piel, Nicholas. ZeroMQ an introduction. ZeroMQ. [En línea] 2010. [Citado el: 10 de Marzo de 2015.] http://nichol.as/zeromq-an-introduction.

57. Symfony. Estándares de codificación. [En línea] 2013. [Citado el: 10 de Marzo de 2015.] http://gitnacho.github.io/symfony-docs-es/contributing/code/standards.html.

58. Quintero, Naychel Pérez de Medina. Sistema Informático para el registro contable de los Útiles y Herramientas de la División Territorial Copextel Matanzas. Matanzas : Universidad de Matanzas "Camilo Cienfuegos", 2013.

59. Puebasdesoftware. Pruebas de Software. Gestión de Calidad y Pruebas de Software. [En línea] [Citado el: 8 de 12 de 2014.] http://pruebasdesoftware.com/pruebadeaceptacion.htm.

60. Beck, Kent, Molina, Jesús García y Aguilar, Luis Loyanes. Una explicación de la programación extrema: Aceptar el cambio. s.l. : Addison Wesley, 2002. ISBN 8478290559.