Universidad de las Ciencias Informáticas

Facultad 1

UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

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

TÍTULO: SISTEMA PARA LA GESTIÓN DE LA INFORMACIÓN DE LA PRÁCTICA PROFESIONAL EN LA FACULTAD

Autora:

Stephanie González González

Tutores:

Msc. PA. Niurvis Legrá Pérez

Ing. As. Mayleidis López Fernández

Ing. Osay González Fuentes

LA HABANA, 2019

DEDICATORIA

Dedico todo mi trabajo a mi madre Janeth González y a mis abuelos José González y Aida Almeida por estar presentes en cada paso de mi carrera y mi vida y haber hecho de mi quien soy ahora. A mis tíos Jaqueline González y René Carranza y a mi prima Elizabeth, por su preocupación y sus exigencias para dar lo mejor de mí en cada momento. A Diosito, mi padre allá en el cielo, que siempre me guía por el mejor camino o al menos lo intenta, porque aun sabiendo lo cabezona que soy, sé que no pierde las esperanzas en mí, como también sé que siempre está mi lado, a él es quien más agradezco, sobre todo por tener a mi familia apoyándome siempre, ¡a todos GRACIAS!!!.

DECLARACIÓN DE AUTORÍA

Declaro por este medio que yo Stephanie González González, con carné de identidad 96111109937 soy la autora principal del trabajo titulado “SISTEMA PARA LA GESTION DE LA INFORMACION DE LA PRACTICA PROFESIONAL EN LA FACULTAD” y autorizo a la Universidad de las Ciencias Informáticas a hacer uso de la misma en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

Para que así conste firmamos la presente a los ____ días del mes de _____ de _____.

_____

Stephanie González G.

Firma de la autora

_____ _____

Msc. Niurvis Legrá Pérez *Ing. Mayleidis López Fernández*

Firma de la Tutora Firma de la Tutora

_____

Ing. Osay González Fuentes

Firma del Tutor

"Las fuerzas productivas, debido al desarrollo continuo del trabajo maquínico, desmultiplicado por la revolución informática, van a liberar una cantidad cada vez mayor del tiempo de actividad humana potencial. Pero ¿Con qué fin? ¿El del paro, la marginalidad opresiva, la soledad, la ociosidad, la angustia, la neurosis, o el bien de la cultura, la creación, la investigación, la reinvención del entorno, el enriquecimiento de los modos de vida y de sensibilidad?"

RESUMEN

La **práctica profesional** es la disciplina principal integradora de la carrera de Ingeniería en Ciencias Informáticas de la Universidad de las Ciencias Informáticas (UCI) la cual concreta el carácter de ciencia de la carrera. Esta disciplina está compuesta por cinco asignaturas fundamentales, divididas en dos ciclos, el básico y el profesional; en este último los estudiantes se encuentran incorporados a proyectos productivos. Durante la práctica profesional cada facultad debe tener control y seguimiento de los estudiantes pertenecientes a los centros productivos y de investigación mientras cursan las asignaturas de dicha disciplina. Este proceso genera mucha información que se encuentra aislada, lo que dificulta ofrecer de forma rápida y oportuna cualquier dato o reporte referente al mismo, además trae consigo pérdidas y en ocasiones duplicidad de la información. Por lo cual el objetivo de la presente investigación es desarrollar un sistema que contribuya a que el intercambio de información entre los centros y la Facultad sea en tiempo y sin pérdidas. El proceso de desarrollo estuvo guiado por la metodología de software "Proceso Unificado Ágil" (AUP por sus siglas en inglés) en su versión UCI, implementando tecnologías y herramientas de software libre como por ejemplo Symfony en el marco de trabajo. El sistema permitió una mejor realización del seguimiento y control de los estudiantes en los Proyectos de Investigación y Desarrollo.

Palabras clave: centros de desarrollo, disciplina integradora, gestión de información, práctica profesional.

INDICE

INDICE

INDICE FIGURAS

INDICE TABLAS

INTRODUCCIÓN

El Plan de Estudios de la carrera Ingeniería en Ciencias Informáticas se concretó a partir del Plan de Estudios C' de Ingeniería Informática. Al fundarse la Universidad de las Ciencias Informáticas (UCI) en el 2002 como parte del Ministerio de la Informática y las Comunicaciones dando respuesta a un conjunto de necesidades del desarrollo social, comenzó un proceso curricular propio marcado por algunas premisas que hicieron que de esta forma se definiera el Plan de Estudios D de Ingeniería en Ciencias Informáticas. Este impulsa a la formación de profesionales que compulsen el desarrollo de la informatización de la sociedad y de la Industria Cubana de Software y Servicios Informáticos. El mismo se estructura en: Modelo del Profesional, Plan del Proceso Docente, Programas de las Disciplinas e Indicaciones Metodológicas y de Organización. (Ministerio de Educación Superior, 2014)

Dentro del Modelo del Profesional se encuentra fundamentado todo lo relacionado con los objetivos de la carrera de Ingeniería en Ciencias Informáticas. El Plan del Proceso Docente de la distribución por semestre y años de estudio y las Indicaciones Metodológicas y de Organización de la Carrera va orientado a la práctica laboral, trabajo independiente del estudiante, estrategias curriculares y evaluación para la culminación de estudios. Todo ello en conjunto da respuesta a las ideas del Comandante en Jefe Fidel Castro, al definirla como "universidad productiva" en 2004 y que implica la necesidad de integración de sus procesos fundamentales: la formación, la producción y la investigación. Fidel puntualizó el carácter de ciencia de la carrera, el cual se concreta a partir de la afirmación de que lo investigativo es esencia de lo académico y de lo laboral al definir como disciplina principal integradora la práctica profesional. (Ministerio de Educación Superior, 2014)

La actividad formativa de la Universidad está compuesta por dos ciclos: el ciclo de integración básico (cinco primeros semestres) y ciclo de integración profesional (cinco últimos semestres). El básico se hace énfasis en la formación académica y tiene como objetivo fundamental la formación del estudiante en las diferentes asignaturas mientras el profesional tiene como objetivo primordial concretar el desempeño de los estudiantes en situaciones profesionales reales. Durante el ciclo profesional el estudiante se enfrenta a una práctica profesional que lo pone en condiciones similares a los entornos laborales en los que se desempeñará una vez egresado.

INTRODUCCIÓN

La actividad de desarrollo-producción de la UCI se soporta sobre la Red de Centros, integrada por 14 centros de desarrollo, un Centro de Soporte, Dirección de Calidad de Software, y diferentes áreas y líneas de investigación que brindan servicios transversales a la producción. *Ver Anexo 1*. En la actualidad el proceso para la inserción de estudiantes en la práctica profesional se realiza a través de una encuesta. En dicha encuesta expresan el interés de pertenecer a cualquiera de estos centros de desarrollo, áreas, líneas de investigación y dirección de investigación. A partir del índice académico y las necesidades de cada área, se realiza la distribución de todas las facultades. Esto evidencia que existen muchos estudiantes de diferentes facultades distribuidos en varias áreas. Para presenciar mejor la problemática se decidió tomar como variable a la Facultad 1 donde se analizaron los siguientes datos obtenidos a través de una entrevista a la Vicedecana de Formación. *Ver Anexo 2.*

La Facultad 1 cuenta con un total de 263 estudiantes entre 3ero, 4to y 5to año, los cuales pertenecen al ciclo profesional. De ese total el 52.5% radican en los 3 centros de la facultad, lo que significa que el 47.14% del resto están ubicados en las áreas restantes. También cuenta con profesores de Proyecto de Investigación y Desarrollo (PID) para atender tanto a los estudiantes de su Facultad como a los estudiantes de otras facultades que residen en los centros de esta, ellos deben estar informados de la situación de cada uno de los estudiantes como asistencia, evaluaciones sistemáticas, desempeño y corte evaluativo. En cada área existe un especialista que es el encargado de centralizar toda la información de los estudiantes a partir de lo circulado por el Supervisor Evaluador Tutor (SET), para luego enviarla a los profesores de la PID en cada facultad. Se encuentra estipulado que cada 15 días se debe entregar un reporte a todas las facultades con dicha información para ser analizada a su debido tiempo, pero a raíz de este proceso se complejiza el cumplimiento de estos reportes. Al existir una mala gestión del seguimiento y control de la práctica profesional, no siempre se realizan los análisis en tiempo con los estudiantes que obtienen resultados desfavorables por lo cual en ocasiones no se toman las medidas para evitar la reincidencia de estas malas prácticas. Todo esto ocasiona una pérdida de tiempo y de información al no existir una forma en la que este documento esté disponible para que así se pueda consultar cuando sea necesario.

En la UCI para la gestión de las evaluaciones se utiliza el Sistema de Gestión Académica (Akademos), herramienta que mantiene informados a los estudiantes y profesores sobre el desempeño académico: matrícula de estudiantes, desarrollo del proceso docente, notas, asistencia, planes de estudio, asignaturas

a cursar, disciplina y perfiles. Este sistema soluciona las dificultades para la gestión de información de las diferentes asignaturas que conforman el Plan de Estudio de la Universidad, no siendo de la misma manera para la asignatura PID debido a su concepción y cómo se maneja en cada Facultad pues lo único que registra es la nota final de la asignatura. En resumen, no existe un sistema desde el cual poder gestionar toda la información acerca de la asignatura Proyecto de Investigación y Desarrollo (PID) de los estudiantes, lo que provoca entre otras cosas atraso en la entrega de reportes y falta de información del proceso realizado.

Teniendo en cuenta la **situación problemática** mencionada anteriormente y la necesidad de encontrar una solución a estas dificultades, el presente trabajo estará dirigido a resolver el siguiente **problema de investigación:** ¿Cómo contribuir al intercambio de información de la Práctica Profesional entre los centros y la facultad?

Enmarcando el **objeto de estudio** el proceso de gestión de la información académica. Enfocando el **campo de acción** en la gestión de la información de la Práctica Profesional en la Facultad.

Se ha propuesto como **objetivo general** desarrollar un sistema que contribuya a que el intercambio de información entre los centros y la Facultad sea en tiempo y sin pérdidas de información.

Objetivos específicos:

- Analizar los referentes teóricos metodológicos fundamentales asociados a los sistemas de gestión de información académica.
- Diseñar el Sistema para la gestión de la información de la Práctica Profesional.
- Implementar las funcionalidades del Sistema para la gestión de la información de la Práctica Profesional.
- Verificar la calidad del Sistema para la gestión de la información de la Práctica Profesional.

Para dar solución a la situación problemática que se presenta, se han propuesto las siguientes **tareas de investigación**:

- Elaboración del estado del arte asociado a la gestión de información académica.
- Identificación de características y concepción de las asignaturas de la PID desde 3ero hasta 5to año con profesores o personal relacionado con la misma.

- Descripción de herramientas y tecnologías informáticas que se usarán en el análisis y diseño del Sistema para la gestión de la información de la Práctica Profesional.
- Selección de la metodología de desarrollo de software que garantice la calidad del Sistema para la gestión de la información de la Práctica Profesional.
- Realización del levantamiento de requisitos para desarrollar el Sistema para la gestión de la información de la Práctica Profesional.
- Modelación del diseño del Sistema para la gestión de la información de la Práctica Profesional.
- Implementación de una primera versión del Sistema para la gestión de la información de la Práctica Profesional con el uso de las tecnologías seleccionadas.
- Realización de las pruebas de calidad al Sistema para la gestión de la información de la Práctica Profesional.

Métodos de la investigación científica empleados:

Métodos teóricos:

- **Histórico-Lógico:** Este método permitió estudiar las formas de solución dadas en años anteriores a problemas similares de gestión de información académica y gestión de la información de la práctica profesional.
- **Analítico-Sintético:** Este método facilitó analizar las teorías presentadas en las bibliografías consultadas y extraer los elementos más importantes relacionados con el proceso de gestión de la información de la disciplina práctica profesional, necesarios para dar solución a la problemática.
- **Modelación:** Esté método posibilitó crear abstracciones con el objetivo de investigar la realidad de los sistemas de gestión de información académica y además posibilitará conocer la respuesta de los procesos sin tener que ejecutar los mismos en el mundo real.

Métodos empíricos:

- **Entrevista:** Es una conversación planificada entre el investigador y el entrevistado para obtener información acerca de la práctica profesional, o sea, el enfoque del negocio.

Con el propósito de realizar una descripción detallada, este documento muestra el resultado de la investigación de la siguiente manera:

Capítulo I: "Fundamentación teórica de Sistemas para la gestión de la información académica". El objetivo de este capítulo es abordar los conceptos asociados a la gestión de la información y la práctica profesional, necesarios para la comprensión de lo que se describe en el resto del trabajo. Se estudiaron las herramientas con las que se llevó a cabo el desarrollo del sistema para la gestión de la información de la práctica profesional en la Facultad, las cuales deben cumplir un conjunto de políticas establecidas, y la definición de la metodología a utilizar.

Capítulo II: "Propuesta de solución para la gestión de la información de la práctica profesional". En este capítulo se enunciaron y describieron las características que presenta el Sistema para la gestión de la información de la práctica profesional. Se describió el objeto de automatización, así como la propuesta que dio solución a los problemas que dieron paso a la investigación. Se realizó el moldeamiento del negocio y se definieron los requisitos funcionales y no funcionales que presenta dicho sistema.

Capítulo III: "Implementación y Pruebas del Sistema para la gestión de la información de la Práctica Profesional". En este capítulo se reflejaron los principales aspectos de la implementación de la solución informática propuesta en el Capítulo II, así como las pruebas realizadas al Sistema para la gestión de la información de la Práctica Profesional. Además, se exponen los resultados de las mismas que permitieron comprobar el correcto funcionamiento del sistema propuesto y verificar que cumple con las necesidades planteadas inicialmente. Asimismo, se expuso lo relacionado con la implementación de una primera versión del Sistema para la gestión de la información de la Práctica Profesional, donde solo se efectúan los casos de uso significativos para la arquitectura. Por otra parte, quedó elaborado un diagrama de despliegue donde se muestran los nodos necesarios para el funcionamiento de la aplicación y un diagrama de componentes donde se presentan los componentes que participan en la implementación.

CAPITULO I: “Fundamentación Teórica del Sistema para la Gestión de la Información de la Práctica Profesional”

En el presente capítulo se expone la caracterización de la práctica profesional, también se fundamenta a través de diferentes conceptos los sistemas de gestión de la información académica y la propia práctica profesional. Además, se realiza un estudio de los sistemas vinculados al campo de acción tanto a nivel internacional como al nacional, se analizan las tecnologías y herramientas que se emplean en la construcción del Sistema para la gestión de la información de la Práctica Profesional.

1.1 Conceptos asociados

1.1.1 Sistema de Gestión de la Información

Gestión de la información (GI) es la denominación convencional de un conjunto de procesos por los cuales se controla el ciclo de vida de la información, desde su obtención (por creación o captura), hasta su disposición final (su archivo o eliminación). Tales procesos también comprenden la extracción, combinación, depuración y distribución de la información a los interesados. El objetivo de la GI es garantizar la integridad, disponibilidad y confidencialidad de la información. (Gestion de información, 2017)

La definición de sistema de información que dan Andreu, Ricart y Valor (1991). Según estos autores, el sistema de información: *Es el conjunto formal de procesos que, operando sobre una colección de datos estructurada de acuerdo con las necesidades de una empresa, recopila, elabora y distribuye la información necesaria para la operación de dicha empresa y para las actividades de dirección y control correspondientes, apoyando, al menos en parte, los procesos de toma de decisiones necesarios para desempeñar las funciones de negocio de la empresa de acuerdo con su estrategia.*

1.1.2 Práctica Profesional

La práctica profesional suele constituirse como el primer paso de un estudiante o de un recién graduado en el mercado laboral. Se trata de una etapa que combina cuestiones típicas de un empleo (la necesidad de alcanzar un cierto grado de productividad, la obligación de acatar las órdenes de un superior, etc.) con elementos más vinculados a la formación y al aprendizaje. (Pérez Porto, 2015)

La práctica profesional es la experiencia desarrollada en un área de la actividad humana rentada. Puede hacer referencia a innumerables oficios u ocupaciones. Así, por ejemplo, es común hablar de la práctica profesional en la medicina, en el derecho, en las ingenierías, etc. La práctica profesional solo puede

desarrollarse trabajando, es la experiencia acumulada de situaciones reales que se presentan en la vida real antes que un conocimiento forjado en los libros. (Editorial Definición MX., 2014)

1.2 Caracterización de la práctica profesional

La actividad investigativa-laboral se fundamenta en el desarrollo de la disciplina de Práctica Profesional que constituye la Disciplina Principal Integradora. La misma está compuesta por 10 asignaturas en segundo, tercero, cuarto y quinto año, con 2048 horas en total, a ejecutarse de forma distribuida durante los semestres involucrados y con la garantía de la incorporación de los estudiantes a proyectos productivos reales. Además, en tercer año se desarrolla en el primer semestre la asignatura Metodología de la Investigación Científica, en el segundo semestre de cuarto año, Componente Profesional de Ingeniería y Gestión de Software (CPIGSW) y durante el segundo semestre de quinto año el Trabajo de Diploma. En cada una de las asignaturas de Proyecto de Investigación y Desarrollo (PID), se han definen los objetivos educativos e instructivos que se deben alcanzar, así como el contenido y las habilidades correspondientes que deben ser desarrolladas. Los responsables en cada Centro de Desarrollo, que existen en cada facultad, se subordinan metodológicamente al Jefe de Departamento de Ingeniería de Software (IGSW) en la Facultad. La Práctica Profesional, de igual forma, se integra y responde a los cortes C-1, (Resolución no. 146/11), a realizarse en las semanas 8 y 16 de cada semestre. Cada estudiante es atendido por un Supervisor Evaluador Tutor (SET), los cuales pueden ser: profesores o especialistas. El profesor de PID de conjunto con los tutores elaboran un plan de formación a los estudiantes que atienden, donde se incluyen actividades relacionadas con la docencia, investigación y producción. En el plan se incluyen las fechas de cumplimiento de las actividades asignadas.

El régimen de trabajo de los estudiantes se ajusta al régimen de cada centro de desarrollo o del grupo de investigación al cual se vinculen. Cada estudiante debe tener al menos una tarea asignada. Las tareas, en los casos que así procedan, deben tener determinados aseguramientos instructivos como: teoría, bibliografía, ejemplos, entre otros. Deben tener además un resultado medible, los cuales pueden ser: artefactos, documentos, informes, aplicación, código fuente, etc. Los resultados de estas tareas deben ser presentados en talleres, seminarios o reuniones de proyectos. La evaluación final de las asignaturas se realiza a través de la discusión de un informe final, donde se presenten los resultados de las tareas realizadas durante la asignatura en todo el semestre. Los tribunales que evalúan la defensa de los informes finales se organizan por el Jefe de Departamento de la Facultad de conjunto con los centros. La

nota final del estudiante la otorga el profesor de PID a partir de las evaluaciones, parciales y finales y el criterio de los tutores. (Ministerio de Educación Superior, 2014)

1.2.1 Roles

En cada asignatura el estudiante debe desarrollar de forma incremental las habilidades asociadas a los roles correspondientes al proceso de informatización de la sociedad. Este abarca desde el proceso de diagnóstico y transformación de los procesos en las entidades, el desarrollo de software y servicios informáticos hasta las tecnologías de la información. Las habilidades a desarrollar se distribuyen en cada asignatura para que el estudiante vaya venciéndolas de manera gradual y ganando en complejidad a medida que las culmina. En cada asignatura el estudiante desarrolla tareas que responden al nivel de las habilidades asociadas a los roles definidos para el año que curse. Es importante que durante el desarrollo de la Práctica Profesional se tengan en cuenta los principios del modelo de integración de lo académico, lo laboral y lo investigativo, por lo que todos los estudiantes deben ser ubicados en el desempeño de roles de proyectos de informatización reales. Los objetivos de las asignaturas y el año se concretan en los planes de formación de cada estudiante y estos a su vez, están orientados a los requisitos exigidos para la certificación formativa de roles, reflejo de las certificaciones profesionales.

Como complemento a la formación del estudiante, se desarrolla el proceso formativo de certificación de roles (opcional), en el cual el estudiante, a partir de las evidencias acumuladas durante cada semestre en el desempeño de los roles, solicita la certificación. Se convoca a un tribunal donde el estudiante presenta sus evidencias y resultados obtenidos, exponiendo los elementos teóricos, herramientas y técnicas utilizadas en el desempeño de los roles correspondientes. Una vez culminado el ejercicio, el tribunal decidirá si otorga o no la certificación al estudiante. En caso de otorgarla, los niveles de certificación que puede obtener son: básico, intermedio y avanzado. (Ministerio de Educación Superior, 2014)

1.3 Análisis de las soluciones existentes vinculadas al campo de acción de la gestión de la información de la Práctica Profesional en la Facultad

1.3.1 Soluciones existentes a nivel internacional

*Aplicación web para la gestión del conocimiento en las asignaturas de producción y realización de audiovisuales. (*Universidad Politécnica de Valencia. Dpto. de Comunicación Audiovisual, Documentación e Historia del Arte (DCADHA)*)*:

CAPÍTULO I

La aplicación web que se ha creado facilita el proceso de búsqueda, extracción y almacenamiento del conocimiento, según las características propias de la asignatura y del tipo de información audiovisual que en ella se maneja, también facilita que los alumnos puedan publicar sus resultados de forma autónoma. Esta aplicación utiliza protocolos estándar de Internet, como HTTP (*Hypertext Transfer Protocol*) y XML (*Extensible Markup Language*), más otros especializados para retransmitir medios *streaming* como MMS (*Microsoft Media Server protocol*) y UDP (*User Datagram Protocol*). La aplicación está programada con ASP (*Active Server Pages*) en el lado del servidor, y con JavaScript en el lado del Cliente. La misma no sólo hace de interfaz entre los usuarios y los registros de una base de datos, sino que marca la forma de aproximarse a la información y cómo utilizarla. En la base de datos se guarda información sobre los documentos de imagen y vídeo que están contenidos en ficheros almacenados en diferentes lugares de la red. Concretamente las imágenes se encuentran en el mismo servidor Web, mientras que los vídeos y sonidos se hallan en otro equipo servidor, especializado en la retransmisión *streaming* de audiovisuales. (Muñoz García, y otros, 2002)

Ventajas:

- Gestiona las tareas realizada por los estudiantes en sus prácticas.
- Permite que toda la documentación que se guarda en su base de datos sea consultada por estudiantes y profesores.

Limitaciones:

- No permite evaluar ninguna de las tareas realizadas, no gestiona las evaluaciones del estudiante en su práctica.
- No es multiplataforma, aunque posee dos servidores para dividir las tareas de la gestión de imágenes y videos.
- No responde a las características de la práctica laboral realizada en la universidad.

Aplicación web para la gestión de entornos virtuales. (Universidad Complutense de Madrid. Facultad de Informática):

Esta es una aplicación para la gestión de entornos virtuales sobre las que se realizarán las prácticas de las asignaturas de una determinada titulación dada. Se distinguen tres tipos o niveles diferentes de

usuarios: alumnos, profesores y administradores. Los cuales tendrán acceso a diferentes funcionalidades y recursos en función de su nivel de autenticación en el sistema. La aplicación consta principalmente de una base de datos desarrollada en lenguaje Ruby con SQLite, donde se almacena toda la información referente a la aplicación y cuatro interfaces o páginas web. El sistema usa el patrón Modelo-Vista-Controlador. Actualmente la aplicación es sencilla y funcional, pero quizás no demasiado interactiva ni atractiva de cara a un potencial usuario. (Almaraz Hernández, y otros, 2011)

Ventajas:

- Control de roles y permisos para la autenticación.
- Permite gestión de la información.

Limitaciones:

- No responde a las características de la práctica laboral realizada en la Universidad.

Aplicación web para la gestión de calificaciones de alumnos. (Universidad Carlos III de Madrid. Departamento de Ingeniería Telemática):

La aplicación Web permite a los docentes de un colegio, instituto o universidad gestionar las calificaciones de sus alumnos para cada curso académico. Teniendo la información de las asignaturas, los grupos y los alumnos organizada. Varios profesores pueden conectarse simultáneamente sin compartir la información. Se permite que un profesor comparta esta información asignando el control de forma parcial a otros profesores. La aplicación permite calificar a los alumnos de una asignatura manualmente o mediante el cálculo automático de las mismas para el caso de calificaciones que dependen del cálculo de una fórmula. Los listados de las calificaciones, así como los datos de las asignaturas o alumnos se pueden importar y/o exportar a un formato reconocible por el usuario. (Núñez Mayorga, 2009)

Ventajas:

- Permite la gestión de las evaluaciones de los estudiantes en las diferentes asignaturas.
- Control de roles y permisos de autenticación.

Limitaciones:

- No responde a las características de la práctica laboral realizada en la Universidad.

1.3.2 Soluciones existentes a nivel nacional

Sistema de Gestión de Información para la educación superior (SiGIFI). (Universidad de Camagüey ¨Ignacio Agramonte Loynaz¨. Facultad de Informática (FI)):

SiGIFI es una aplicación web que cuenta con una interfaz amigable y fácil de utilizar. Este sistema está basado en la tecnología Cliente-Servidor, lo cual permite una mayor velocidad en el funcionamiento de la aplicación. El mismo fue desarrollado haciendo uso de la tecnología Apache-PHP-MySQL y el lenguaje de programación en PHP, actualmente, este lenguaje se puede ejecutar bajo Apache, IIS, entre otros. PHP generalmente es utilizado como módulo de Apache, lo que lo hace ser rápido. La eficiencia de este sistema permite generar toda la información y la documentación, extensión universitaria y de ciencia y técnica de la FI. (Sistema de Gestión de Información para la Educación Superior, 2014)

Ventajas:

- Control de documentos recibidos o generados.
- Generar informes docentes y reportes.
- Publicación de resultados académicos.
- Archiva toda la información referente a docencia a través de un Sistema de Gestión Integral de Documentos de Archivos v2.0.
- Es multiplataforma.

Limitaciones:

- No responde a las características de la práctica laboral realizada en la Universidad.

Web docente para la práctica profesional I de la Ingeniería Informática, en la semipresencialidad. (Universidad Tecnológica de La Habana José Antonio Echeverría (CUJAE)):

El sitio Web docente tiene un diseño estructural no lineal y en los diferentes menús se brinda información de la Disciplina Práctica Profesional y de la asignatura Práctica Profesional I como: programa analítico,

secuencia de actividades, indicaciones, artículos del reglamento docente relacionados con la Práctica Profesional, bibliografías, entre otras. Su implementación se realizó utilizando el Sistema de Gestión de Contenidos (CMS) Drupal versión 6.27 y WAMP versión 2.2. (de la Torre Matamoro, 2013)

Ventajas:

- Informa sobre todo lo referente a la práctica profesional de la carrera, actividades, artículos, etc.

Limitaciones:

- No es multiplataforma.
- Es solamente informativo.
- No responde a las características de la práctica laboral realizada en la Universidad.

Sistema para la gestión de la Práctica Profesional en la Facultad 3. (UCI):

El sistema elaborado permite realizar el control y seguimiento de los estudiantes en los Proyectos de Investigación y Desarrollo, facilitando un conjunto de reportes que agilizan la entrega de la información que se solicite. Se utilizó la metodología de software Variación de AUP para la Universidad y en la implementación se emplearon tecnologías y herramientas de software libre. El mismo aporta de manera significativa a la informatización y mejora de los procesos de control y seguimiento de los estudiantes pertenecientes a los centros de desarrollo, mientras cursan las asignaturas PID de la disciplina de Práctica Profesional en el ciclo profesional. (Bagarotti Abreu, 2017)

Ventajas:

- Tecnología reciente y adecuada a los requisitos del centro de desarrollo.
- Realiza reportes sobre la información generada en el desarrollo de la práctica.
- Gestiona la información de los estudiantes en la práctica profesional como son las evaluaciones y tareas a realizar.

Limitaciones:

- No gestiona los proyectos desarrollados por los estudiantes en la asignatura PID.

- No responde a las características de la práctica laboral realizada en la Universidad en la actualidad, ya que fue realizada en un momento en el que el desarrollo de la práctica profesional se realizaba en los propios centros de la facultad y además existían departamentos de la práctica en las facultades.

Soluciones a tener en cuenta para el desarrollo del sistema:

Gestor de proyectos (Gespro). (UCI):

Xedro Gespro es una Suite orientada a la web que permite la planificación, seguimiento y control de productos en forma de proyectos. Cuenta con herramientas para el apoyo a la toma de decisiones a nivel de proyectos, nivel de entidad ejecutora y nivel gerencial. Se presenta en un modelo de negocios basado en servicios que combinan el uso de una solución informática para la dirección integrada de proyectos y un sistema de formación especializada en gestión de proyectos. Esta combinación posibilita no solo la informatización de las organizaciones sino también la mejora integral de los proyectos de planificación, control y seguimiento de proyectos. (UCI, 2013)

Ventajas:

- Puede gestionar las tareas de los estudiantes.
- Puede guardar los proyectos desarrollados durante la práctica profesional.

Limitaciones:

- No gestiona las evaluaciones de los estudiantes.
- Se debe pedir permiso en cada área para que los profesores tengan acceso a las evaluaciones y proyectos de sus estudiantes.

Sistema de Gestión académica (Akademos). (UCI):

En la UCI para la gestión de las evaluaciones se utiliza el Sistema de Gestión Académica (Akademos), herramienta que mantiene informado a los estudiantes y profesores sobre el desempeño académico: matrícula de estudiantes, desarrollo del proceso docente, notas, asistencia, planes de estudio, asignaturas a cursar, disciplina y perfiles. Este sistema soluciona las dificultades para la gestión de información de las

diferentes asignaturas que conforman el Plan de Estudio de la Universidad, no siendo de la misma manera para la asignatura PID debido a su concepción y cómo se maneja en cada facultad pues lo único que registra es la nota final de la asignatura.

Ventajas:

- Está organizado de manera que cada profesor tiene acceso al grupo que debe atender cualquiera sea su facultad, sus evaluaciones, asistencias, etc.
- Gestiona las evaluaciones y la asistencia de los estudiantes en cada una de las asignaturas cursadas.

Limitaciones:

- No lleva planificación de actividades productivas.
- No gestiona las tareas desarrollada por los estudiantes en la práctica profesional.
- No registra el plan de formación de los estudiantes en la práctica profesional.

Conclusiones de las soluciones existentes

Luego de la investigación realizada sobre las soluciones existentes tanto nacionales como internacionales se llegó a la conclusión de realizar un sistema que permita el mejoramiento del flujo de información de la práctica profesional. Los sistemas encontrados no brindan todos los requisitos que se deben tener en cuenta para resolver los problemas existentes en la gestión de la práctica profesional. Las ventajas detectadas constituyen una fuente inicial para la definición de requisitos y las funcionalidades que no pueden faltar en el sistema para la gestión de la información de la práctica profesional. Además, a partir de la profunda indagación de las características de estos sistemas se observó que en las universidades analizadas no se realiza la práctica profesional de igual manera. La mayoría de dichos sistemas no se encuentran disponibles o no se llegaron a implementar y quedaron solo en la fase teórica de un proyecto o tesis, lo que indica la necesidad de un nuevo sistema que responda a las características de la práctica realizada en la Universidad de las Ciencias Informáticas.

CAPÍTULO I

1.4 Metodología de desarrollo de software

Encontrar una definición estándar para una metodología de desarrollo de software es complicado, pero es posible encontrar algunas de las utilizadas por compañías que se dedican a esta labor. Ejemplo es la de *Centers for medicare & medicaid services* o *CMS* (2017), quienes la definen como *un marco de trabajo que es utilizado para estructurar, planificar y controlar el proceso de desarrollo de un sistema de información,* por otro lado en la revista electrónica Internacional *Journal of Computer Aplictions* (Chandra, 2015) la define como *un proceso mediante el cual un proyecto de software es completado o desarrollado a través de procesos o etapas bien definidas.*

Metodología de desarrollo de software Variación de AUP para la UCI

Toda metodología debe ser adaptada a las características de cada proyecto es por eso que no existe una metodología universal, por lo cual la universidad decidió hacer una variación de la metodología ágil AUP para que se adaptara al ciclo de vida definido para la actividad productiva de la UCI. Esta metodología está definida por el centro de estudios como documento rector de la actividad productiva. (Rodríguez Sánchez, 2014)

Descripción de las Fases AUP-UCI

De las 4 fases que propone AUP (Inicio, Elaboración, Construcción, Transformación) la adaptación define para el ciclo de vida de los proyectos de la UCI mantener la fase de Inicio, pero modificando el objetivo de la misma. Se unifican las 3 restantes fases en una sola llamándola Ejecución, agregándole una nueva fase de Cierre. AUP propone 7 disciplinas (Modelo, Implementación, Prueba, Despliegue, Gestión de configuración, Gestión de proyectos y Entorno), esta define para el ciclo de vida de los proyectos de la UCI tener 8 disciplinas, pero a un nivel más atómico que el definido en AUP (Rodríguez Sánchez, 2014) para mejor comprensión se muestra la Tabla 1.

Tabla 1. Disciplinas de la Variación AUP-UCI (Rodríguez Sánchez, 2014)

Disciplinas Variación AUP-UCI	Objetivos Disciplinas
Modelado de Negocio (opcional)	El Modelado del Negocio es la disciplina

	destinada a comprender los procesos de negocio de una organización. Se comprende cómo funciona el negocio que se desea informatizar para tener garantías de que el software desarrollado va a cumplir su propósito.
Requisitos	El esfuerzo principal en esta disciplina es desarrollar un modelo del sistema que se va a construir. Esta comprende la administración y gestión de los requisitos funcionales y no funcionales del producto.
Análisis y diseño	En esta disciplina, si se considera necesario, los requisitos pueden ser refinados y estructurados para conseguir una comprensión más precisa de estos, y una descripción que sea fácil de mantener y ayude a la estructuración del sistema (incluyendo su arquitectura). Además, en esta disciplina se modela el sistema y su forma para que soporte todos los requisitos, incluyendo los no funcionales. Los modelos desarrollados son más formales y específicos que el de análisis.
Implementación	En la implementación, a partir de los resultados del Análisis y Diseño se construye el sistema.
Pruebas interna	En esta disciplina se verifica el resultado de la implementación probando cada construcción, incluyendo tanto las construcciones internas como intermedias, así como las versiones finales a ser liberadas. Se deben desarrollar artefactos

	de prueba como: diseños de casos de prueba, listas de chequeo y de ser posibles componentes de prueba ejecutables para automatizar las pruebas.
Pruebas de liberación	Pruebas diseñadas y ejecutadas por una entidad certificadora de la calidad externa, a todos los entregables de los proyectos antes de ser entregados al cliente para su aceptación.
Pruebas de adaptación	Es la prueba final antes del despliegue del sistema. Su objetivo es verificar que el software está listo y que puede ser usado por usuarios finales para ejecutar aquellas funciones y tareas para las cuales fue construido.
Despliegue (opcional)	Constituye la instalación, configuración, adecuación, puesta en marcha de las soluciones informáticas y entrenamiento personal del cliente.

Escenarios para la disciplina Requisitos

Se parte del modelado de negocio, el mismo propone 3 variantes: Casos de Uso del Negocio (CUN), Descripción del proceso del negocio (DPN) o Modelo Conceptual (MC). Además, existen 3 formas de encapsular los requisitos: Casos de usos (CUS), Historias de usuarios (HU), Descripción de requisitos por procesos (DRP) (Rodríguez Sánchez, 2014). De allí que surgen 4 escenarios para modelar el sistema en los proyectos, se mantiene en 2 de ellos el MC, queda de la siguiente forma:

Escenario No 1: Proyectos que modelen el negocio con CUN solo pueden modelar el sistema con CUS.

Escenario No 2: Proyectos que modelen el negocio con MC solo pueden modelar el sistema con CUS.

Escenario No 3: Proyectos que modelen el negocio con DPN solo pueden modelar el sistema con DRP

Escenario No 4: Proyectos que no modelen negocio solo pueden modelar el sistema con HU.

Características por escenarios

Escenario No 1: Aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan que puedan modelar una serie de interacciones entre los trabajadores del negocio/actores del sistema (usuario), similar a una llamada y respuesta respectivamente, donde la atención se centra en cómo el usuario va a utilizar el sistema. Es necesario que se tenga claro por el proyecto que los CUN muestran como los procesos son llevados a cabo por personas y los activos de la organización.

Escenario No 2: Aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan que no es necesario incluir las responsabilidades de las personas que ejecutan las actividades, de esta forma modelarían exclusivamente los conceptos fundamentales del negocio. Se recomienda este escenario para proyectos donde el objetivo primario es la gestión y presentación de información.

Escenario No 3: Aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan un negocio con procesos muy complejos, independientes de las personas que los manejan y ejecutan, proporcionando objetividad, solidez, y su continuidad. Se debe tener presente que este escenario es muy conveniente si se desea representar una gran cantidad de niveles de detalles y la relaciones entre los procesos identificados.

Escenario No 4: Aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan un negocio muy bien definido. El cliente estará siempre acompañando al equipo de desarrollo para convenir los detalles de los requisitos y así poder implementarlos, probarlos y validarlos. Se recomienda en proyectos no muy extensos, ya que una HU no debe poseer demasiada información.

Por lo tanto, al tener un negocio bien definido, estar con el cliente durante todo el proceso de desarrollo del proyecto para la toma de decisiones, no ser un proyecto extenso y definir historias de usuarios, la autora ha decidido utilizar la metodología AUP variación UCI en su escenario número 4 (HU).

1.5 Herramientas, lenguajes y tecnologías

Lenguaje de modelado de software

UML (Unified Modeling Language. Lenguaje unificado de modelado)

Es un estándar diseñado para visualizar, especificar, construir y documentar software orientado a objetos. Proporciona un medio de visualizar la organización de alto nivel de los programas sin fijarse con detenimiento en los detalles del código real. El lenguaje de modelado está unificado porque está basado en varios modelos previos (métodos de Booch, Rumbaugh y Jacobson).

La parte más importante de UML, al menos a un nivel de iniciación o medio en programación, reside en un conjunto rico de diagramas gráficos. Diagramas de clases que muestran las relaciones entre clases, diagramas de objetos que muestran cómo se relaciona objetos específicos entre sí, diagramas de casos de uso que muestran cómo los usuarios de un programa interactúan con el programa. UML no es un proceso de desarrollo de software sino, simplemente, un medio para examinar el software que se está desarrollando. Aunque, al ser un estándar, UML se puede aplicar a cualquier tipo de lenguaje de programación. (Object Management Group, 2005)

CAPÍTULO I

Herramientas para el modelado del sistema

VP 8.0 (Visual Paradigm)

Visual Paradigm para UML es una herramienta para desarrollo de aplicaciones utilizando modelado UML ideal para Ingenieros de Software, Analistas de Sistemas y Arquitectos de sistemas que están interesados en construcción de sistemas a gran escala y necesitan confiabilidad y estabilidad en el desarrollo orientado a objetos. *Visual Paradigm for UML Enterprise Edition (VP-UML EE)*: Es la edición top de la línea de productos, lo que representa todo lo más moderno y agrega valor en términos de modelado de datos orientado a objetos, hace posible la documentación del proyecto, mapeo relacional de objetos para Java, .NET y PHP, reduciendo costos y aumentando su productividad. (Huanca Velarde, 2014)

Esta herramienta fue seleccionada por sus características y, además, porque la Universidad cuenta con una licencia otorgada para su utilización, igualmente tiene una gran importancia por el uso del software libre en Cuba y por las características que posee.

Lenguajes de programación

HTML 5 (*HyperText Markup Language*, Lenguaje de Marcado de Hipertextos)

Es el lenguaje básico de casi todo el contenido web. Brinda opciones para incluir hojas de cálculo y otras aplicaciones directamente en sus documentos. Además, permite recuperar información en línea a través de enlaces de hipertexto y diseñar formularios para realizar transacciones mediante servicios remotos como es la búsqueda de información. (Romero, 2009)

CSS 3 (*Cascading Style Sheets,* Hojas de Estilo en Cascada)

Se utiliza para definir el aspecto de todos los contenidos, como: el color, tamaño y tipo de letra de los párrafos de texto, la separación entre titulares y párrafos, la tabulación con la que se muestran los elementos de una lista. Esta forma de descripción de estilos ofrece a los desarrolladores el control total sobre estilo y formato de sus documentos. (MDN webs docs, CCS3, 2016)

PHP 7.2 (*HyperText Preprocessor,* Preprocesador de Hipertexto)

Es un lenguaje de programación muy popular utilizado especialmente para la creación de páginas web dinámicas. Es de código abierto y distribuido bajo la licencia PHP, lo que hace posible reutilizar o modificar el código fuente sin costes adicionales. La función básica de PHP es leer datos de formularios y convertirlos en variables PHP. Posteriormente, las variables pueden, por ejemplo, ser introducidas en una

base de datos o enviadas por correo electrónico. Las ventajas de PHP incluyen su integración con el protocolo de Internet y un amplio soporte de diferentes modelos de bases de datos. Todos los códigos PHP se procesan en el lado del servidor. Esto significa que los scripts PHP se ejecutan directamente, a diferencia de otros códigos de programación como JavaScript, que se ejecutan del lado de quien solicita el acceso. (Desarrollo Web; PHP7: más velocidad de carga y acceso en Internet, 2018) En resumen, las novedades más importantes son:

- Mejoras en el rendimiento, casi el doble que la versión 5.6.
- Bajos requerimientos de RAM.
- Implementación de un árbol sintáctico abstracto para la generación de códigos.
- Soporte constante de sistemas de 64 bits.
- Mejora en la gestión de errores, los errores conducen raramente a caídas del sistema.
- Generador de números aleatorios criptográficamente más seguros.
- Eliminación de extensiones e interfaces del servidor antiguas y desactualizadas.
- Definición de los tipos de datos escalares y tipos de devolución de códigos.

JavaScript 1.2

Es un lenguaje de programación que puede utilizarse para añadir interactividad a las páginas Web. A menudo se llama a JavaScript "lenguaje de escritura de guiones" (lenguaje de *scripting*) denotando que es más fácil escribir guiones que programar. En este caso, es una distinción sin diferencia. Un guión Script es un programa que está contenido internamente dentro de una página HTML. JavaScript permite crear una interfaz de usuario activa, lo que ofrece retroalimentación a los visitantes según navegan por las páginas.
Se puede utilizar JavaScript para asegurarse de que los usuarios introducen información válida en los formularios, lo que dará como resultado ahorro en tiempo. (Delgado, 2018)

Herramientas y tecnologías

PhpStorm 2.5.0

Es un Entorno de Desarrollo Integrado (IDE) compatible con PHP desde su versión 5.3 hasta la 7.0, proporciona prevención de errores, mejor autocompletado y refactorización de código, depuración de configuración cero, y un editor HTML, CSS, JavaScript. El IDE ofrece completamiento inteligente de código, resaltado de sintaxis, configuración extendida del formato de código, comprobación de errores,

plegado de código, soporta las mezclas de idiomas, entre otros. Esta herramienta se seleccionó para el desarrollo de la propuesta de solución ya que responde a la política de soberanía tecnológica de la Universidad. (Delgado, 2018)

Marco de trabajo

Symfony 3.4

Es un *framework* PHP de tipo *full-stack* construido con varios componentes independientes creados por el proyecto Symfony. (Symfony, 2015)

Principales características:

- Su código, y el de todos los componentes y librerías que incluye, se publican bajo la licencia MIT (*Massachusetts Institute of Technology*) de software libre.
- La documentación del proyecto es libre e incluye varios libros y decenas de tutoriales específicos.
- Aprender a programar con Symfony permite acceder a una gran variedad de proyectos el *framework* Symfony3.4 para crear aplicaciones complejas, el micro *framework Silex* para sitios web sencillos y los componentes Symfony para otras aplicaciones PHP.
- La seguridad es tan importante para el proyecto Symfony, que antes de su lanzamiento, se encargó una auditoría de seguridad a una empresa independiente.
- La herramienta Capifony (basada en el proyecto Capistrano y creada por miembros de la comunidad Symfony) simplifica el *deploy* de las aplicaciones Symfony, incluso en múltiples servidores y bases de datos.
- La excelente herramienta Composer, que simplifica de forma radical la instalación y gestión de las dependencias de las aplicaciones PHP, también ha sido creada por varios miembros de la comunidad Symfony.

Se decide hacer uso de este *framework* por:

- Fácil de instalar y configurar en la mayoría de plataformas.
- Independiente del sistema gestor de bases de datos ya que su capa de abstracción permite cambiar con facilidad de sistema gestor de base de datos en cualquier fase del proyecto.

- Utiliza programación orientada a objetos, de ahí que sea imprescindible PHP 5.
- Sencillo de usar en la mayoría de casos, está más indicado para grandes aplicaciones Web que para pequeños proyectos.

Servidor de aplicaciones web

Apache 2.4.29

Es el servidor web hecho por excelencia, su configuración, robustez y estabilidad hacen que cada vez millones de servidores reiteren su confianza en este programa. (Mifsuf Talón, 2012)

El servidor Apache es una tecnología gratuita y de código abierto que se distingue por las siguientes características:

- Puede ser ejecutado en una multitud de Sistemas Operativos, lo que lo hace prácticamente universal.
- Altamente configurable de diseño modular. Es muy sencillo ampliar sus capacidades.
- Permite personalizar la respuesta ante los posibles errores que se pueden dar en el servidor.

Gestor de base de datos

PostgreSQL 9.4

Es un potente sistema de base de datos relacional de objetos de código abierto con más de 30 años de desarrollo activo que le ha ganado una sólida reputación de confiabilidad, solidez de funciones y rendimiento. PostgreSQL se ejecuta en todos los principales sistemas operativos, ha sido compatible con ACID (atomicidad, consistencia, aislamiento, durabilidad) desde 2001, y tiene complementos poderosos como el popular extensor de base de datos geoespacial PostGIS. Esta versión agrega muchas nuevas características que mejoran la flexibilidad, escalabilidad y rendimiento de PostgreSQL para diferentes tipos de usuarios de bases de datos, incluyendo mejoras al soporte para JSON, replicación y rendimiento de los índices. (PostgreSQL 9.4, 2014)

1.6 Patrón Arquitectónico Modelo Vista Controlador

El MVC o Modelo-Vista-Controlador es un patrón de arquitectura de software que, utilizando 3 componentes separa la lógica de la aplicación de la lógica de la vista en una aplicación, esto significa que

cuando se hace un cambio en alguna parte del código, esto no afecte otra parte del mismo. Por ejemplo, si se modifica la Base de Datos, sólo se debe modificar el modelo que es quién se encarga de los datos y el resto de la aplicación debería permanecer intacta. Esto respeta el principio de la responsabilidad única. Es decir, una parte de tu código no debe de saber qué es lo que hace toda la aplicación, sólo debe de tener una responsabilidad. Es una arquitectura importante puesto que se utiliza tanto en componentes gráficos básicos hasta sistemas empresariales. (Alvares, 2014)

Modelo

Se encarga de los datos, generalmente (pero no obligatoriamente) consultando la base de datos, actualizaciones, consultas, búsquedas, etc.

Controlador

Se encarga de controlar, recibe las órdenes del usuario y se encarga de solicitar los datos al modelo y de comunicárselo a la vista.

Vistas

Son la representación visual de los datos. Ni el modelo, ni el controlador se preocupan de cómo se verán los datos, esta responsabilidad es únicamente de la vista.

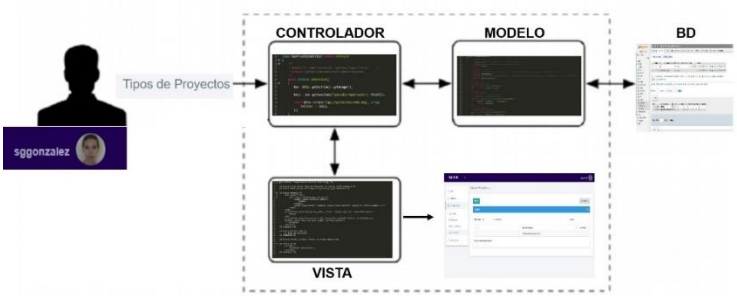


Figura 1: Arquitectura Modelo Vista Controlador. Elaboración propia.

Conclusiones del capítulo

El análisis del estado del arte de los principales conceptos asociados al dominio del problema permitió una mejor comprensión del proceso que siguen los sistemas de gestión de la información académica y cómo se realiza la práctica profesional. Los sistemas de gestión de la información académica analizados

permitieron conocer que ciertas funcionalidades no pueden faltar en la solución a desarrollar. Las ventajas presentadas en el análisis de los sistemas homólogos constituyen una fuente inicial para la definición de requisitos. En el desarrollo de la aplicación se emplea la metodología AUP-UCI, la herramienta *case Visual Paradigm for UML* para modelar el sistema y los lenguajes de programación PHP para el lado del servidor y como lenguajes del lado del cliente HTML5, CSS3 y JavaScript1.8.5. Se utilizará el *framework* de desarrollo Symfony3.4.

CAPITULO II: “Propuesta de solución para la gestión de la información de la Práctica Profesional”

En el presente capítulo se exponen las principales características del Sistema para la gestión de la información de la Práctica Profesional, o sea el análisis y el diseño de la propuesta de solución. Se exponen los conceptos fundamentales del dominio. Se evidencian las necesidades del cliente, los requisitos que dicho sistema debe cumplir y se muestran las historias de usuario, el plan de iteraciones y otros artefactos generados por la metodología durante el desarrollo de la solución.

2.1 Propuesta de solución

Para dar respuesta a la problemática planteada se decidió realizar un Sistema para la gestión de la información de la PP. Este sistema cuenta con 3 módulos Administración, Configuraciones y Mi área. El módulo de Administración cuenta con 5 nomencladores ellos son:

Administrar Facultades: se crean las facultades a las que pertenecen los estudiantes.

Administrar grupos: a partir de una facultad seleccionada de las creadas con anterioridad se crean los grupos a los que pertenecen los estudiantes.

Administrar Áreas: se crean las áreas de investigación y desarrollo de la Universidad en donde se realizan las prácticas de los estudiantes, una vez creadas se les puede asignar diferentes proyectos a partir de un tipo de proyecto definido.

Administrar Estudiante: insertando un usuario a través de un buscar accediendo al servicio LDAP se ubica el nombre del estudiante y su solapín. Luego se puede agregar a la lista de estudiantes. Posteriormente, se les debe asignar de los grupos ya creados el que le corresponde a cada uno de ellos y, además, se les debe asignar también el área a la que pertenecen para realizar su PP.

Administrar Personas: insertando un usuario a través de un buscar accediendo al servicio LDAP se ubica el nombre del trabajador y su solapín. A continuación, se procede a agregar a la lista de personas. Una vez insertados se les debe asignar un rol, que puede ser Administrador, Jefe de área, Jefe de PID de la facultad, Profesor de PID o se deja el pre-definido por el sistema que sería usuario, y además se le designará el área a la que pertenecen.

El módulo de Configuraciones cuenta con 3 nomencladores ellos son:

Tipo de Proyecto: se crean tipos de proyectos para que puedan ser utilizados en la creación de proyectos y en el plan de formación.

Sesiones de Trabajo: se crean las sesiones de trabajo que se utilizan en otras funcionalidades.

Rol Estudiante: se crean los diferentes roles del estudiante que les son asignados luego para su trabajo en las diferentes áreas, como por ejemplo programador y analista.

El módulo de Mi Área de acuerdo al rol que sea asignado cambia en el caso del Jefe del área, el Profesor de PID y el Administrador cuenta con3 niveles que poseen diferentes funcionalidades ellos son:

Plan de Formación: se crea un plan de formación para los diferentes estudiantes en donde se tienen en cuenta la asignatura que se imparte, en este caso las diferentes PID; la cantidad de horas semanales, cantidad de horas en general, la fecha de realizado y los diferentes estudiantes a los que se les realiza este plan de formación. Las tareas que deben ejecutar los estudiantes se insertan al plan en cuanto se creen en otra funcionalidad. Este plan de formación se puede exportar en Pdf para su mejor distribución a estudiantes y profesores que participen en el negocio a través de un reporte.

Mis Estudiantes: se observa la lista de los estudiantes que pertenecen al centro del usuario que se encuentre logueado en ese momento. El usuario puede asignarles las tareas a los estudiantes que luego se adicionan al Plan de Formación. Deben definir la fecha de inicio, cantidad de horas, fecha de cumplimiento y descripción de la tarea. Una vez creada en el listado de tareas de ese estudiante aparece como “no cumplida”. Una vez realizada la tarea por parte de este y entregada a su profesor o jefe de área, se debe editar la tarea y cambiar por “cumplida” para posteriormente evaluarla. A partir del tipo de nota, ya sea pregunta escrita o evaluación final, por ejemplo, se le asigna el valor de la nota obtenida donde se guarda en un listado de notas del estudiante.

Mis Proyectos: se listan los proyectos que se hayan creado con anterioridad y se pueden crear nuevos. Además, se asigna a los estudiantes y trabajadores que pertenecen a los mismos. En este módulo se puede pasar asistencia por proyecto a los diferentes estudiantes que pertenezcan al centro o área de investigación introduciendo datos como la fecha, la sesión en que se encuentra y si se encuentra ausente o presente, en el caso de ser ausencia si es justificada. Estas asistencias se podrán exportar a un Pdf para su mejor distribución a estudiantes y profesores que participen en el negocio a través de un reporte.

En caso de los estudiantes Mi Área solamente mostrará:

Mis Datos: donde se encuentra la asistencia del estudiante en el transcurso del semestre, un listado de sus notas y tareas y podrá ver su perfil, nombre, usuario, grupo y rol. No se permite modificar.

Plan de Formación: el estudiante puede observar el plan de formación definido con anterioridad por su profesor o jefe de área.

CAPÍTULO II

El rol de Administrador tiene acceso a todo el sistema mientras que el rol de Estudiante, Jefe de área y Profesor de PID, así como el rol pre-definido por el sistema tiene solamente acceso a Mi Área, y el Jefe de PID de la facultad solamente accede al módulo de Administración, para asignar los estudiantes a las diferentes áreas entre otras funcionalidades.

2.2 Proceso de negocio

Luego de la distribución de los estudiantes de 3ero, 4to y 5to año en las diferentes áreas de la Universidad para realizar la práctica profesional, se les asigna un tema de investigación a desarrollar y los supervisores, evaluadores y tutores (SET). Estos estudiantes son atendidos además por un profesor que de conjunto con los SET elaboran los planes de formación y los evalúa. Es importante tener en cuenta la ubicación del estudiante en el puesto laboral, ya sea en el proyecto al que está vinculado o en un laboratorio de producción de los diferentes centros. A partir de este proceso el profesor de la PID de la facultad mantiene el control y seguimiento de los estudiantes en las asignaturas de PID tanto de los pertenecientes a dicha facultad como de los que realizan sus prácticas en los centros de la facultad, pero no pertenecen a la misma.

2.2.1 Modelo conceptual

Los conceptos y relaciones de la solución se muestran en la figura 2, donde el estudiante es el actor principal de la PP. Cada uno de ellos pertenece a un área, puede o no ser de la facultad, donde se le asigna un tema de investigación (dependiendo del año académico que cursa, para los estudiantes de 5to año corresponde el tema de tesis). Los trabajadores que se encuentran relacionados directamente con los estudiantes pueden ser profesores, especialistas o recién graduados en adiestramiento (RGA), en el que uno o más será el tutor de uno o varios estudiantes.

El modelo se elabora para utilizarlo como base y definir las restricciones y funcionalidades que el sistema debe cumplir. A partir de este modelo y del análisis realizado en el capítulo anterior de los sistemas homólogos se determinan los requisitos que deben satisfacer la problemática abordada, definidos en el próximo epígrafe en la disciplina Requisitos según la metodología de desarrollo seleccionada.

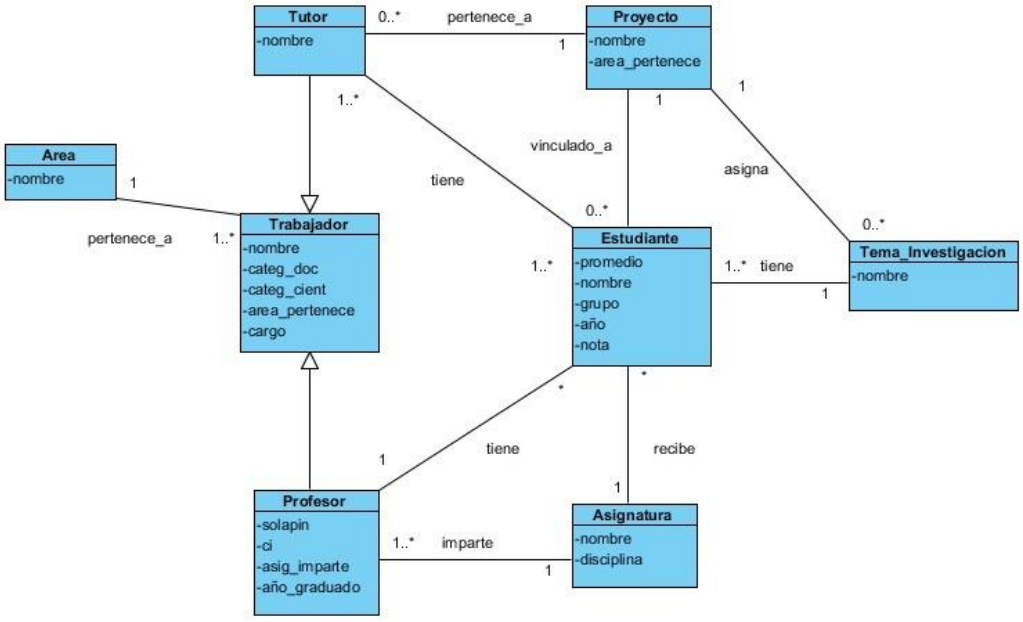


Figura 2. Modelo conceptual. Elaboración Propia

2.3 Requisitos

El esfuerzo principal en esta disciplina es desarrollar un modelo del Sistema para la gestión de la información de la Práctica Profesional. Esta comprende la administración y gestión de los requisitos funcionales y no funcionales del producto. (Rodríguez Sánchez, 2014). Es una característica que el sistema DEBE tener o es una restricción que el sistema DEBE satisfacer para ser aceptada por el cliente. Levantamiento de requisitos es la especificación del sistema en términos que el cliente entienda, de forma que se constituya en el contrato entre el cliente y los desarrolladores. (Quiroga, 2012)

2.3.1 Fundamentos para la obtención de requisitos y técnicas de identificación de requisitos

Las fuentes de obtención de requisitos a consultar para la definición de los requisitos del sistema son los siguientes:

- Soluciones informáticas existentes con propósitos similares a los deseados.
- Modelo conceptual.
- Expertos de la Facultad 1.

2.3.2 Técnicas utilizadas para la obtención de requisitos

Entrevista: La entrevista se puede definir como un "intento sistemático de recoger información de otra persona" a través de una comunicación interpersonal que se lleva a cabo por medio de una conversación estructurada. Debe quedar claro que no basta con hacer preguntas para obtener toda la información necesaria. Es muy importante la forma en que se plantea la conversación y la relación que se establece en la entrevista. (Guerra, 2016) *Ver Anexo 3*

Análisis documental: La documentación ya sea manual o reportes proporciona información valiosa con respecto a organizaciones y sus operaciones, es difícil que refleje la forma en la que se desarrollan realmente las actividades o donde se encuentra el poder de la toma de decisiones, pero es de gran importancia para introducir al analista al dominio de operación y el vocabulario que utiliza. (Guerra, 2016) *Ver sub-epígrafe 1.4*

Construcción de prototipos: Los prototipos de sistema permiten a los usuarios experimentar para ver cómo éste ayuda a su trabajo. Fomentan el desarrollo de ideas que desembocan en requisitos. Además de permitir a los usuarios mejorar las especificaciones de requisitos. En general, el uso de esta técnica es un medio que permite solventar objeciones del usuario del tipo: "No sé exactamente lo que quiero, pero lo sabré cuando lo vea". Por lo general, la construcción de prototipos incrementa los costos en las etapas iniciales de un proyecto, pero esto se recupera en etapas posteriores gracias al mejor entendimiento de los requisitos por parte de los desarrolladores. En algunos casos también se utiliza como un medio para formalizar la aceptación previa del cliente de los requisitos del proyecto. (Guerra, 2016) *Ver sub-epígrafe 2.4.6*

2.3.3 Requisitos funcionales

Los requisitos funcionales (RF) describen la interacción entre el sistema y su ambiente independientemente de su implementación. El ambiente incluye al usuario y cualquier otro sistema externo que interactúa con el sistema. (Quiroga, 2012)

RF	Nombre	Complejidad
RF1	Gestionar estudiantes	Alta
RF1.1	Listar estudiantes	Alta
RF1.2	Adicionar estudiantes	Alta
RF1.3	Modificar estudiantes	Alta
RF1.4	Eliminar estudiantes	Alta
RF2	Gestionar trabajador	Alta
RF2.1	Listar trabajador	Alta
RF2.2	Adicionar trabajador	Alta
RF2.3	Modificar trabajador	Alta
RF2.4	Eliminar trabajador	Alta
RF3	Gestionar asistencia	Media
RF3.1	Listar asistencia	Media
RF3.2	Adicionar asistencia	Media
RF3.3	Modificar asistencia	Media
RF4	Gestionar notas	Media
RF4.1	Listar notas	Media
RF4.2	Adicionar notas	Media
RF4.3	Modificar notas	Media
RF4.4	Eliminar notas	Media
RF5	Gestionar tareas	Alta
RF5.1	Listar tareas	Alta

RF5.2	Adicionar tareas	Alta
RF5.3	Modificar tareas	Alta
RF5.4	Eliminar tareas	Alta
RF6	Gestionar plan de formación	Alta
RF6.1	Listar plan de formación	Alta
RF6.2	Adicionar plan de formación	Alta
RF6.3	Modificar plan de formación	Alta
RF6.4	Eliminar plan de formación	Alta
RF7	Generar reportes	Media
RF8	Autenticar usuario	Media
RF9	Gestionar Proyecto	Alta
RF9.1	Adicionar Proyecto	Media
RF9.2	Listar Proyecto	Media
RF9.3	Modificar Proyecto	Media

2.3.4 Requisitos no funcionales

Describen aspectos del sistema que son visibles por el usuario que no incluyen una relación directa con el comportamiento funcional del sistema. Los requisitos no funcionales (RNF) incluyen restricciones como el tiempo de respuesta (desempeño), la precisión, recursos consumidos, seguridad, etc. (Quiroga, 2012). Los requisitos no funcionales definidos son:

RNF. 1 Usabilidad

RNF. 1.1 La arquitectura del sistema debe permitir que con al menos 2 clics en la página de inicio el usuario pueda acceder a la funcionalidad deseada.

RNF. 1.2 La arquitectura del sistema debe contar con un máximo de 3 módulos o niveles de acceso para mejor uso del sistema.

RNF. 2 Confiabilidad y Seguridad

RNF. 2.1 El sistema concede acceso a cada usuario autenticado solo a las funcionalidades que le estén permitidas de acuerdo al rol que presenten, garantizando que la información no sea expuesta a personal indebido.

RNF. 3 Disponibilidad

RNF. 3.1 El sistema debe brindar servicios las 24h los 7días de la semana

RNF. 3.2 El sistema debe permitir la navegación de múltiples usuarios al mismo tiempo.

RNF. 4 Rendimiento

RNF. 4.1 Los tiempos de respuesta y velocidad de procesamiento de la información son menores de 5seg.

RNF. 5 Verificación de datos

RNF. 5.1 El sistema debe validar la información que se introduzca en cualquiera de los formularios. Esta validación incluye la obligatoriedad de los campos, el tipo y sus características.

RNF. 6 Hardware

RNF. 6.1 Para un correcto funcionamiento la estación de trabajo debe contar con un mínimo de: procesador Pentium IV, RAM de 2GB, disco duro de 50 GB y una tarjeta de red de 100Mb/s.

RNF. 7 Software

RNF. 7.1 El sistema debe poder ser visualizado por los navegadores web Firefox en la versión 58.0 y Chrome en su versión 62.0.3202.75.

2.3.5 Parámetros elegidos para la validación de requisitos

El objetivo principal de la validación de requisitos es que todos estos requisitos estén correctamente definidos pues de lo contrario se corre el riesgo de implementar una mala especificación, con el costo que eso conlleva. Los parámetros a validar en los requisitos son: (MADEJA, 2015)

- **Validez:** No basta con preguntar a un usuario, todos los potenciales usuarios pueden tener puntos de vista distintos y necesitar otros requisitos.
- **Consistencia:** No debe haber contradicciones entre unos requisitos y otros.
- **Completitud:** Deben estar todos los requisitos. Esto es imposible en un desarrollo iterativo, pero, al menos, deben estar disponibles todos los requisitos de la iteración en curso.
- **Realismo:** Se pueden implementar con la tecnología actual.
- **Verificabilidad:** Tiene que existir alguna forma de comprobar que cada requisito se cumple.

Existen diferentes técnicas para la validación de requisitos de software como:

Técnica de prototipado: El prototipado de interfaz de usuario es una técnica de representación aproximada de la interfaz de usuario de un sistema software que permite a clientes y usuarios entender más fácilmente la propuesta de los ingenieros de requisitos para resolver sus problemas de negocio. Los dos tipos principales de prototipos de interfaz de usuario son: (MADEJA, 2015)

- Desechables: se utilizan sólo para la validación de los requisitos y posteriormente se desechan. Pueden ser prototipos en papel o en software.
- Evolutivos: una vez utilizados para la validación de los requisitos, se mejora su calidad y se convierten progresivamente en el producto final.

Revisiones Técnicas Formales: Es una actividad que da garantía a la calidad del software, los objetivos son: (Uninotas, 2016)

- Descubrir errores en la lógica o en la implementación de cualquier representación de software.
- Con los RTF debemos verificar que el software bajo una revisión alcanza los requisitos funcionales.
- Garantizar que el software ha sido desarrollado de acuerdo a los estándares predefinidos.

- Con esto se consigue un software desarrollado en forma uniforme.
- Con esto se consigue y se persigue que los proyectos sean más manejables.

La validación de los requisitos de software se desarrolló a través de la técnica de prototipado. En este proceso se diseñaron los prototipos de interfaces. A lo largo del desarrollo del proyecto se realizaron diferentes pruebas funcionales al sistema para detectar los posibles errores y evitar su propagación. Esto permitió que el cliente tuviera noción de la propuesta de solución y se pudieran hacer las correcciones necesarias antes del diseño e implementación de las funcionalidades del Sistema para la gestión de la información de la Práctica Profesional.

2.4 Fase de planificación

En la fase de planificación se describen las funcionalidades del sistema detalladamente, empleando HU. Se establecen las prioridades para implementarlas (establecidas por el cliente) y se estima el tiempo de realización de cada una. El resultado de esta fase es un plan de entrega.

2.4.1 Historias de Usuarios

Las Historias de Usuarios sirven para escribir lo que el usuario desea ser capaz de hacer. Además, se centran en el valor que viene de usar el sistema en lugar de una especificación detallada de lo que el sistema debe hacer. Están concebidos como un medio para fomentar la colaboración. A continuación se detallan las HU: (Angel Luis Lozano Sánchez, 2016)

Tabla 2. Plantilla general para las HU. (Reinoso Miranda, et al., 2015)

Historia de Usuario	
No: Posee el número asignado a la HU.	**Nombre:** Atributo que contiene el nombre de la HU.
Usuario: El usuario del sistema que utiliza o protagoniza a la HU.	
Prioridad en Negocio: Evidencia el nivel de	**Iteración Asignada:** Precisa la iteración en la

prioridad de la HU en el negocio.	que será desarrollada la HU.
Riesgo en Desarrollo: Evidencia el nivel de riesgo en caso de no realizarse la HU. (Alto/ Medio/ Bajo)	**Puntos Estimados:** Permite estimar la duración de la implementación. Cuando el valor es uno equivale a una semana de trabajo (5 días trabajando 40 horas, es decir, 8 horas diarias). Por lo que cuando el valor del atributo es de 0.5 equivale a 2 días y medio de trabajo, lo que se traduce en 20 horas.
Descripción: Posee una breve descripción de lo que realizará la HU.	
Observaciones: Brinda información extra que se estime agregar para hacer más comprensible de la HU.	

Tabla 3. Gestionar estudiante. Elaboración propia

Historia de Usuario	
No: 1	**Nombre:** Gestionar estudiantes
Usuario: Profesor principal de la PP, Vicedecana de Formación	
Prioridad en Negocio: Alta	**Iteración Asignada:** 1
Riesgo en Desarrollo: Alto	**Puntos Estimados:** 3
Descripción: Se debe tener la posibilidad de gestionar la información de los estudiantes. La funcionalidad Gestionar estudiantes permite listar, modificar, insertar y eliminar estudiante.	

Observaciones: El usuario debe estar autenticado.

Tabla 4. Gestionar trabajador. Elaboración propia

Historia de Usuario	
No: 2	**Nombre:** Gestionar trabajador
Usuario: Profesor principal de la PP, Vicedecana de Formación	
Prioridad en Negocio: Alta	**Iteración Asignada:** 1
Riesgo en Desarrollo: Alto	**Puntos Estimados:** 3
Descripción: Se debe tener la posibilidad de gestionar la información de los trabajadores. La funcionalidad Gestionar trabajador permite listar, modificar, insertar y eliminar trabajador.	
Observaciones: El usuario debe estar autenticado.	

Tabla 5. Autenticar usuario. Elaboración propia

Historia de Usuario	
No: 3	**Nombre:** Autenticar usuario
Usuario: Profesor principal de la PP, Vicedecana de Formación, estudiantes, SET	
Prioridad en Negocio: Media	**Iteración Asignada:** 1

Riesgo en Desarrollo: Medio	**Puntos Estimados:** 2
Descripción: El usuario debe tener la posibilidad de autenticarse para interactuar con la aplicación.	
Observaciones: Para el acceso al sistema es necesario estar autenticado.	

Tabla 6. Gestionar tarea. Elaboración propia

Historia de Usuario	
No: 4	**Nombre:** Gestionar tareas
Usuario: SET	
Prioridad en Negocio: Alta	**Iteración Asignada:** 2
Riesgo en Desarrollo: Medio	**Puntos Estimados:** 2.5
Descripción: Se debe tener la posibilidad de gestionar las tareas. La funcionalidad Gestionar tareas permite listar, modificar, insertar y eliminar tarea.	
Observaciones: El usuario debe estar autenticado.	

Tabla 7. Gestionar notas. Elaboración propia

Historia de Usuario	
No: 5	**Nombre:** Gestionar notas

Usuario: SET, profesor PID	
Prioridad en Negocio: Media	**Iteración Asignada:** 2
Riesgo en Desarrollo: Medio	**Puntos Estimados:** 2.5
Descripción: Se debe tener la posibilidad de gestionar las notas. La funcionalidad Gestionar notas permite listar, modificar, insertar y eliminar nota.	
Observaciones: El usuario debe estar autenticado.	

Tabla 8. Gestionar asistencia. Elaboración propia

Historia de Usuario	
No: 6	**Nombre:** Gestionar asistencia
Usuario: SET, profesor PID	
Prioridad en Negocio: Media	**Iteración Asignada:** 2
Riesgo en Desarrollo: Medio	**Puntos Estimados:** 2
Descripción: Se debe tener la posibilidad de gestionar la asistencia. La funcionalidad Gestionar asistencia permite listar, modificar, insertar y eliminar asistencia.	
Observaciones: El usuario debe estar autenticado.	

Tabla 9. Gestionar plan de formación. Elaboración propia

Historia de Usuario	
No: 7	**Nombre:** Gestionar plan de formación
Usuario: SET, profesor PID	
Prioridad en Negocio: Alta	**Iteración Asignada:** 2
Riesgo en Desarrollo: Medio	**Puntos Estimados:** 1.5
Descripción: Se debe tener la posibilidad de gestionar el plan de formación. La funcionalidad Gestionar plan de formación permite listar, modificar e insertar el plan de formación. Además, el usuario debe tener la posibilidad de exportar e imprimir el plan de formación.	
Observaciones: El usuario debe estar autenticado.	

Tabla 10. Módulo de reportes. Elaboración propia

Historia de Usuario	
No: 8	**Nombre:** Generar reportes
Usuario: SET, profesor PID, Vicedecana de Formación	
Prioridad en Negocio: Media	**Iteración Asignada:** 2
Riesgo en Desarrollo: Medio	**Puntos Estimados:** 1.5

Descripción: Cada uno de los encargados de la PP debe tener acceso a un resumen del recorrido de los estudiantes en la práctica profesional para poder dar un resultado final en la asignatura, estos reportes deben ser generados cada 15 días. Además, el usuario debe tener la posibilidad de exportar e imprimir los reportes.
Observaciones: El usuario debe estar autenticado.

Tabla 11. Gestionar Proyectos. Elaboración propia

Historia de Usuario	
No: 9	**Nombre:** Gestionar Proyectos
Usuario: Profesor del área	
Prioridad en Negocio: Alta	**Iteración Asignada:** 2
Riesgo en Desarrollo: Alta	**Puntos Estimados:** 1.5
Descripción: Se debe tener la posibilidad de gestionar la información de los proyectos. La funcionalidad gestionar proyecto permite listar, modificar, insertar y eliminar un proyecto.	
Observaciones: El usuario debe estar autenticado.	

2.4.2 Estimación de esfuerzos por historia de usuario

Las estimaciones asociadas a la implementación de las historias de usuario la establecen los programadores utilizando como medida el punto. Un punto equivale a una semana ideal de programación. Los resultados obtenidos se exponen en la presente tabla.

Tabla 12. Estimación de esfuerzo por HU. Elaboración propia

No	Historia de Usuario	Puntos Estimados
1	Gestionar estudiantes	3
2	Gestionar trabajador	3
3	Autenticar usuario	2
4	Gestionar Tareas	2.5
5	Gestionar Notas	2.5
6	Gestionar asistencia	2
7	Gestionar plan de formación	1.5
8	Generar reportes	1.5

2.4.3 Plan de Iteraciones

El siguiente nivel de la planificación es el plan de iteración. Después de identificar cada una de las HU y estimar el esfuerzo dedicado al desarrollo, se procede a la implementación, en este caso se resolvió efectuar en tres iteraciones, detalladas a continuación.

Iteración 1.

En la presente iteración se realizan las historias de usuarios elegidas. HU 1: Gestionar estudiantes

HU 2: Gestionar trabajador

HU 3: autenticar usuario

Se añade la funcionalidad necesaria para gestionar la autenticación y autorización de los usuarios y las funcionalidades relativas a la gestión de estudiantes y gestión del trabajador. La realización de las mismas va dando una idea de cómo queda la aplicación en sus inicios.

Iteración 2.

Esta iteración conlleva a la finalización de la aplicación tras la implementación de las historias correspondientes a la generación de reportes, gestionar plan de formación, gestión de asistencia, gestión de tareas y notas.

2.4.4 Plan de duración de las iteraciones

El plan de duración de las iteraciones se realiza luego de tener el estimado en semanas que demora implementar cada HU. Se tiene en cuenta la prioridad que el cliente le asigna a cada historia y el nivel de complejidad que poseen.

Tabla 13. Plan de duración de iteraciones. Elaboración propia

Iteraciones	Orden de las HU a implementar	Duración
Iteración 1	HU 1: Gestionar estudiantes	8 semanas
	HU 2: Gestionar trabajador	
	HU 3: Autenticar usuario	
Iteración 2	HU 4: Gestionar tareas	10 semanas
	HU 5: Gestionar notas	
	HU 6: Gestionar asistencia	
	HU 7: Gestionar plan de formación	
	HU 8: Generar Reportes	

2.4.5 Plan de entregas

Tabla 14. Plan de entregas. Elaboración propia

Iteración	Fecha de Entrega
1	15 marzo

2	1 Junio

2.4.6 Prototipo no funcional de la interfaz de usuario

Es un modelo del comportamiento del sistema que puede ser usado para entenderlo completamente o ciertos aspectos de él y así clarificar los requisitos. Un prototipo es una representación de un sistema, aunque no es un sistema completo, posee las características del sistema final o parte de ellas. Se utiliza principalmente para probar ciertos aspectos del diseño que no implican dentro de los objetivos del proyecto, puede revelar errores u omisiones en lo requisitos propuestos. (Reyes R., y otros, 2009)

Figura 3. Prototipo no funcional. Pantalla Principal. Elaboración propia

2.5 Modelo de diseño

Describe la manera de comunicarse el software dentro de sí mismos, con sistemas que inter-operan dentro de él y con las personas que lo utilizan. Una interfaz implica un flujo de información y un tipo específico de comportamiento. Define la relación entre los elementos estructurales principales del software, los patrones de diseño que se pueden utilizar para lograr los requisitos y las restricciones que afectan a la manera en que se pueden aplicar los patrones. Es el marco de trabajo del sistema. (Allamandri, 2012)

2.5.1 Diseño arquitectónico

La etapa de diseño es la etapa central en relación con la arquitectura y probablemente la más compleja. Durante esta etapa se definen las estructuras que componen la arquitectura. La creación de estas estructuras se hace en base a patrones de diseño, tácticas de diseño y elecciones tecnológicas. El diseño que se realiza debe buscar ante todo satisfacer los requisitos que influencian a la arquitectura, y no simplemente incorporar diversas tecnologías porque están “de moda”. (Arquitectura de Software, 2018)

A continuación, se presenta la separación de las clases de la aplicación según el marco de trabajo Symfony3.4, visualizando el patrón MVC.

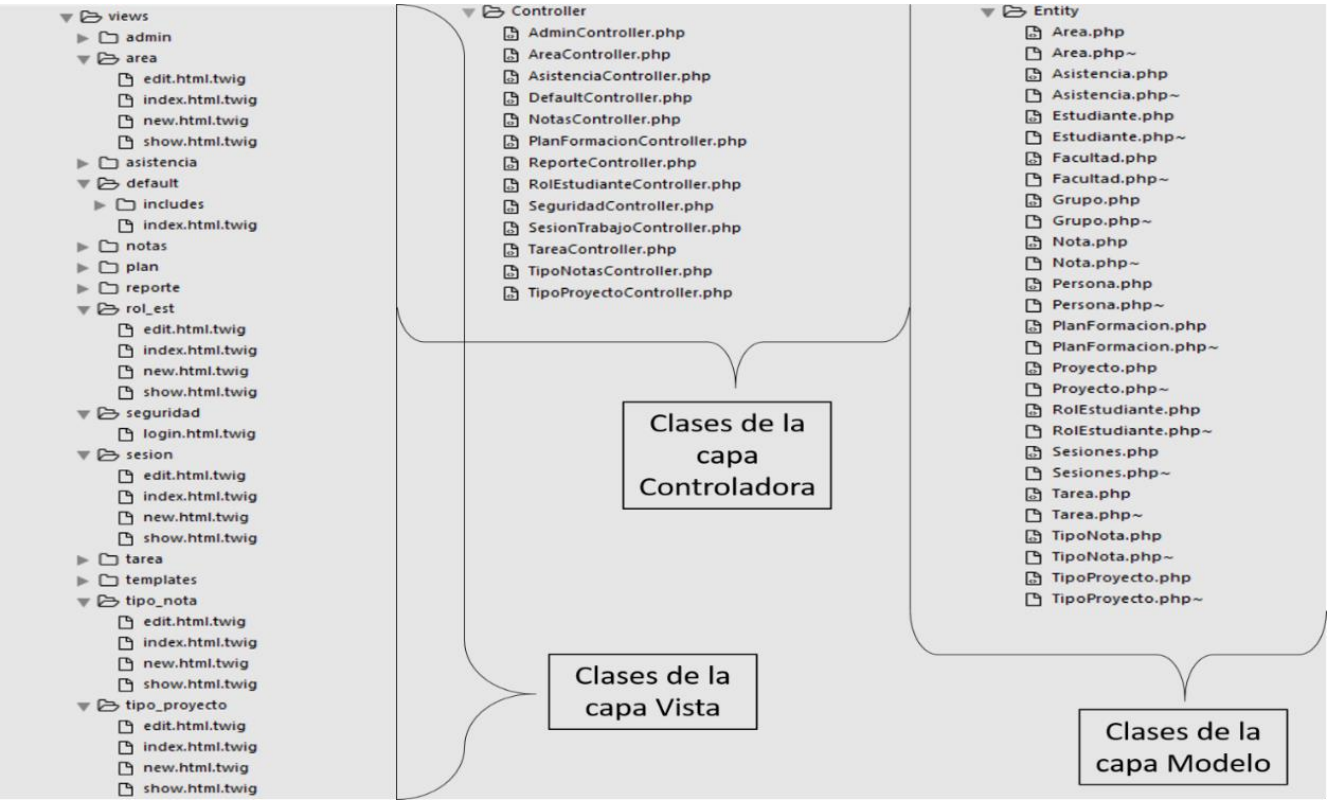


Figura 4. Aplicación del patrón MVC. Elaboración propia

2.5.2 Diagrama de clases

Un diagrama de clases permite representar gráficamente y de manera estática la estructura general de un sistema, mostrando cada una de las clases y sus interacciones (como herencias, asociaciones, etc.). Se representan en forma de bloques, los cuales son unidos mediante líneas y arcos. Los diagramas de clases son el pilar fundamental del modelado con UML, siendo ampliamente utilizados tanto para análisis como para diseño de sistemas y software en general. (Cillero, 2016)

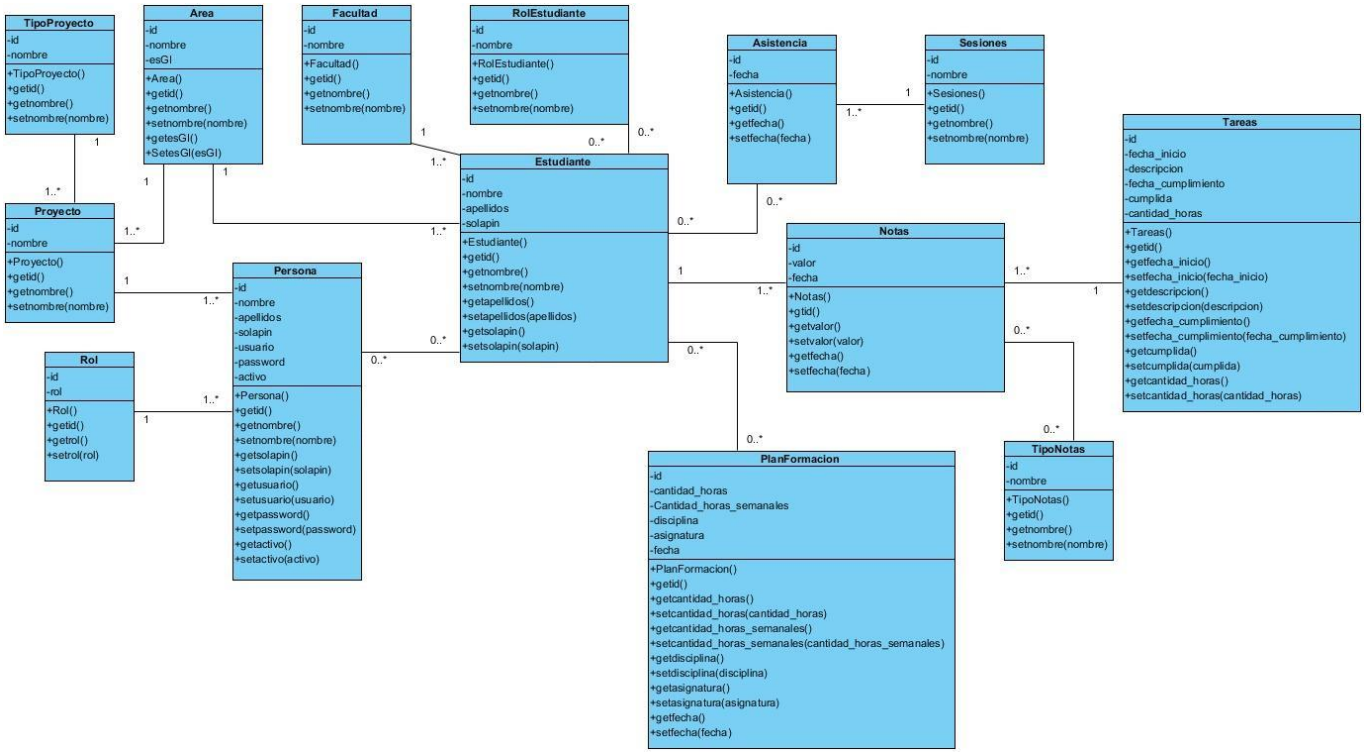


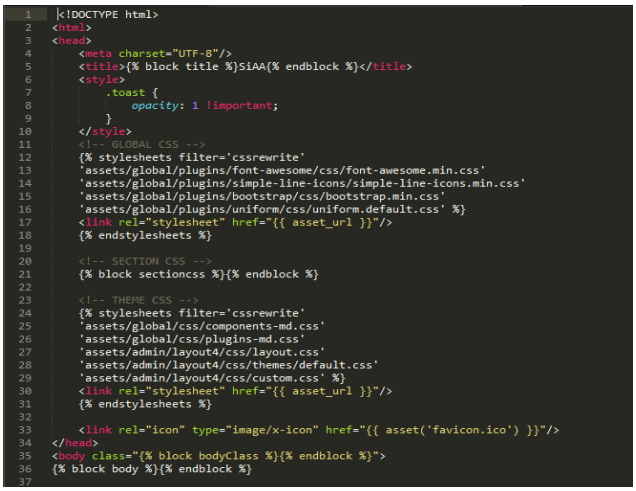
Figura 5. Diagrama de clases. Elaboración propia

2.5.3 Patrones del diseño de software

En la búsqueda de soluciones a los problemas comunes que se presentan en el desarrollo de software se utilizaron patrones de diseños como herramienta de apoyo ya que facilitan la reutilización y la capacidad de expansión del software, reduciendo la complejidad del código y del acoplamiento y facilitan el mantenimiento.

Patrones GoF (Gang of four)

Se clasifican en 3 categorías basadas en su propósito: creacionales, estructurales y de comportamiento. En el diseño de la solución se utilizó el patrón Decorador, contenido dentro de la categoría de los patrones estructurales de GoF. El propósito de este patrón es añadir funcionalidades adicionales a una clase dinámicamente, proporcionando una alternativa flexible a la especialización mediante herencia, aporta una mayor flexibilidad que la herencia estática, permitiendo añadir una funcionalidad dos o más. Este patrón se implementa en la clase *View* de Symfony, que es utilizada en la creación de plantillas para las paginas *html.twig*. (Leiva, 2016)



```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8"/>
    <title>{% block title %}SiAA{% endblock %}</title>
    <style>
        .toast {
            opacity: 1 !important;
        }
    </style>
    <!-- GLOBAL CSS -->
    {% stylesheets filter='cssrewrite'
    'assets/global/plugins/font-awesome/css/font-awesome.min.css'
    'assets/global/plugins/simple-line-icons/simple-line-icons.min.css'
    'assets/global/plugins/bootstrap/css/bootstrap.min.css'
    'assets/global/plugins/uniform/css/uniform.default.css' %}
    <link rel="stylesheet" href="{{ asset_url }}"/>
    {% endstylesheets %}

    <!-- SECTION CSS -->
    {% block sectioncss %}{% endblock %}

    <!-- THEME CSS -->
    {% stylesheets filter='cssrewrite'
    'assets/global/css/components-md.css'
    'assets/global/css/plugins-md.css'
    'assets/admin/layout4/css/layout.css'
    'assets/admin/layout4/css/themes/default.css'
    'assets/admin/layout4/css/custom.css' %}
    <link rel="stylesheet" href="{{ asset_url }}"/>
    {% endstylesheets %}

    <link rel="icon" type="image/x-icon" href="{{ asset('favicon.ico') }}"/>
</head>
<body class="{% block bodyClass %}{% endblock %}">
{% block body %}{% endblock %}
```

Figura 6. Fragmento de código de la clase base.html.twig. Elaboración propia

```
{% extends 'base.html.twig' %}

{% block sectioncss %}
    {% stylesheets filter='cssrewrite'
    'assets/global/plugins/select2/select2.css'
    'assets/global/plugins/bootstrap-toastr/toastr.min.css' %}
    <link rel="stylesheet" href="{{ asset_url }}"/>
    {% endstylesheets %}

    {% block pagecss %}{% endblock %}
{% endblock %}

{% block bodyClass %}
    page-md page-header-fixed
{% endblock %}
```

Figura 7. Ejemplo del uso del patrón decorador. Elaboración propia

Patrones GRASP (General Responsibility Assignment Software Patterns)

Patrones para la asignación de responsabilidades y el fomento de buenas prácticas para el diseño de software. (Leiva, 2016)

Los principales patrones GRASP son:

- Experto: asigna la responsabilidad al experto en información. Se observa el uso de este patrón en todas las clases a utilizar en la solución ya que cada clase conoce su información y es la encargada de implementar las funcionalidades que brindan información de las mismas. (La figura a continuación muestra la clase Estudiante que contiene toda la información referente a los estudiantes)

Estudiante
-id
-$facultad: int
-$area: int
-$usuario: int
-$nombre: string
-$apellidos: string
-$solapin: string

Figura 8. Clase Estudiante. Elaboración propia

- Creador: designa la clase que debe ser responsable de crear una instancia de otra. Este patrón es utilizado en las clases controladoras, en las cuales se crean objetos de las clases entidades para su manipulación. *Ver figura 9.*
- Bajo acoplamiento: este patrón expresa que entre las clases deberán existir pocas ataduras, es decir, estas estarán lo menos relacionadas posible de forma tal que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de las clases. Permite crear clases más independientes y más reutilizables. *Ver figura 5.*
- Alta cohesión: asigna responsabilidades de manera que la cohesión se mantenga alta, de este modo se hace más fácil la comprensión, fácil reutilización, fácil mantenibilidad y mayor resistencia a los cambios. Este patrón se evidencia en el controlador EstudianteController.php en el que si se

realiza un cambio en alguno de sus atributos también se deben modificar los métodos relacionados con los mismos. *Ver figura 9.*

- Controlador: asigna responsabilidades al “controlador”, que será una clase que represente al sistema completo, una parte activa del mundo real que desencadena de tal evento, representa un manejador artificial de eventos. Como ejemplo se muestra la clase EstudianteController.php. *Ver figura 9.*

EstudianteController.php
+indexAction()
+newAction(Request $request)
+showAction(Estudiante $estudiante)
+editAction(Request $request, Estudiante $estudiante)
+deleteAction(Request $request, Estudiante $estudiante)
+createDeleteForm(Estudiante $estudiante)

Figura 9. Clase controladora EstudianteController.php. Elaboración propia

2.5.4 Modelo de datos

Un modelo de datos describe los datos que apoyan los procesos de una organización y refleja con exactitud cómo serán guardados los datos en la base de datos. Contiene cada una de las tablas de la aplicación, así como sus atributos y relaciones. El modelo se encuentra normalizado en Tercera Forma Normal, por lo que todos los atributos de las tablas del modelo, dependen únicamente de la llave primaria.

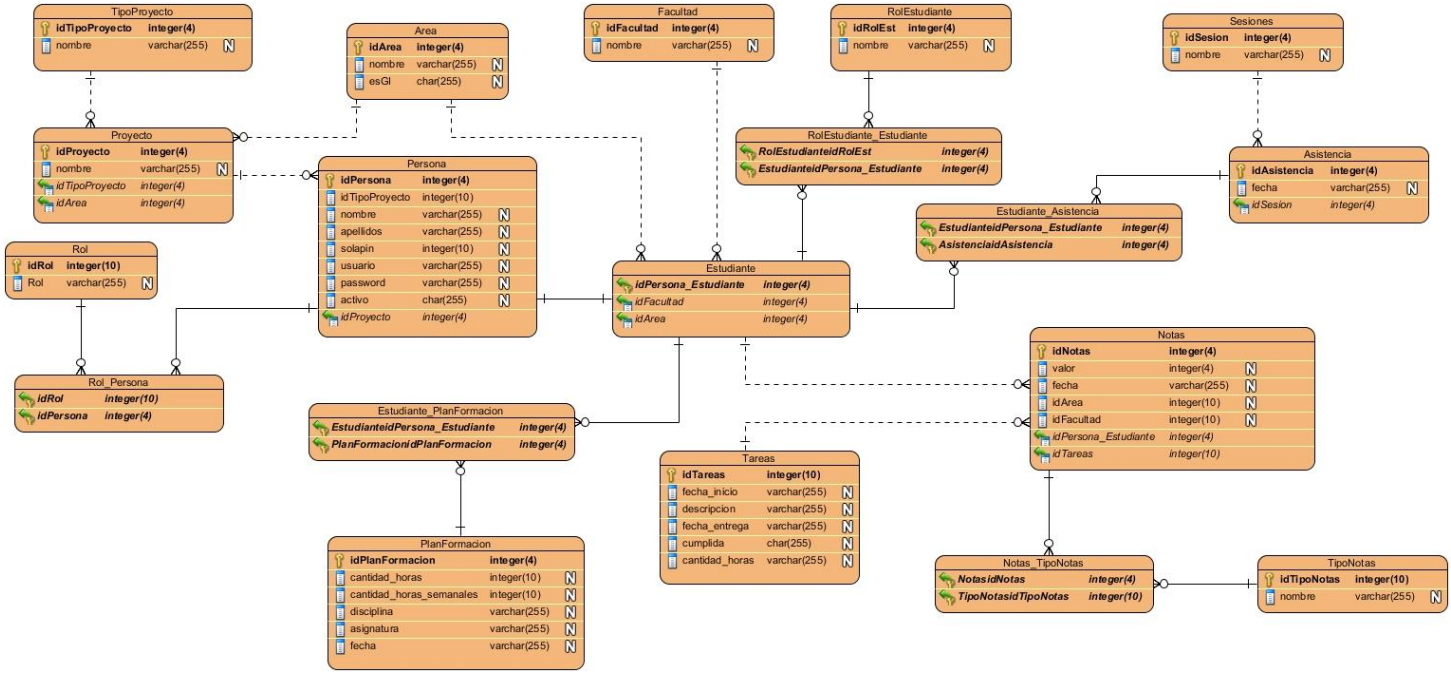


Figura 10. Modelo de datos. Elaboración propia

Conclusiones del capítulo

En este capítulo se abordaron los contenidos de las fases de exploración, planificación y diseño correspondiente a la metodología AUP-UCI con lo cual se desarrolla el Sistema para la gestión de la información de la Práctica Profesional. Lo anterior sirvió como guía para la implementación de dicho sistema. El estudio de los sistemas de gestión de información académica y las entrevistas realizadas permitieron definir 8 requisitos funcionales y 5 requisitos no funcionales. Al definir el modelo de datos se entendió y organizó mejor la información. Al aplicarse los patrones del diseño y arquitectónicos se creó una estructura común y conocida para varios programadores, lo que facilitó un mejor mantenimiento y reutilización del Sistema para la gestión de la información de la Práctica Profesional.

CAPITULO III: “Implementación y Pruebas del Sistema para la gestión de la información de la Práctica Profesional”

En el presente capítulo se reflejan los principales aspectos de la implementación de la solución informática propuesta, las pruebas realizadas al Sistema para la gestión de la información de la Práctica Profesional y los resultados de las mismas. Lo anterior permite comprobar el correcto funcionamiento de dicho sistema y verificar que cumple con las necesidades planteadas inicialmente.

3.1 Modelo de implementación

El modelo de implementación describe cómo los elementos del modelo de diseño se implementan en términos de componentes. Este modelo representa también la organización de los componentes de acuerdo con los mecanismos de estructuración y modulación disponibles en el entorno de implementación y en el lenguaje de programación empleado y cómo dependen los componentes unos de otros. (Guadarrama, Jorge;, 2013)

3.1.1 Diagrama de componentes

Un diagrama de componentes representa cómo un sistema de software es dividido en componentes y muestra las dependencias entre estos. Los diagramas de componentes prevalecen en el campo de la arquitectura de software, pero pueden ser usados para moldear y documentar cualquier arquitectura del sistema. Muestran las dependencias lógicas entre componentes, sean estos fuentes, binarios o ejecutables. (Guadarrama, Jorge;, 2013)

La *Figura 11* presenta el diagrama de componentes de la solución, en él se observan los componentes o unidades físicas de implementación con interfaces bien definidas, que son utilizados como parte reemplazable de un sistema e incorporan la implementación de ciertas clases del diseño del software y las relaciones existentes entre ellos.

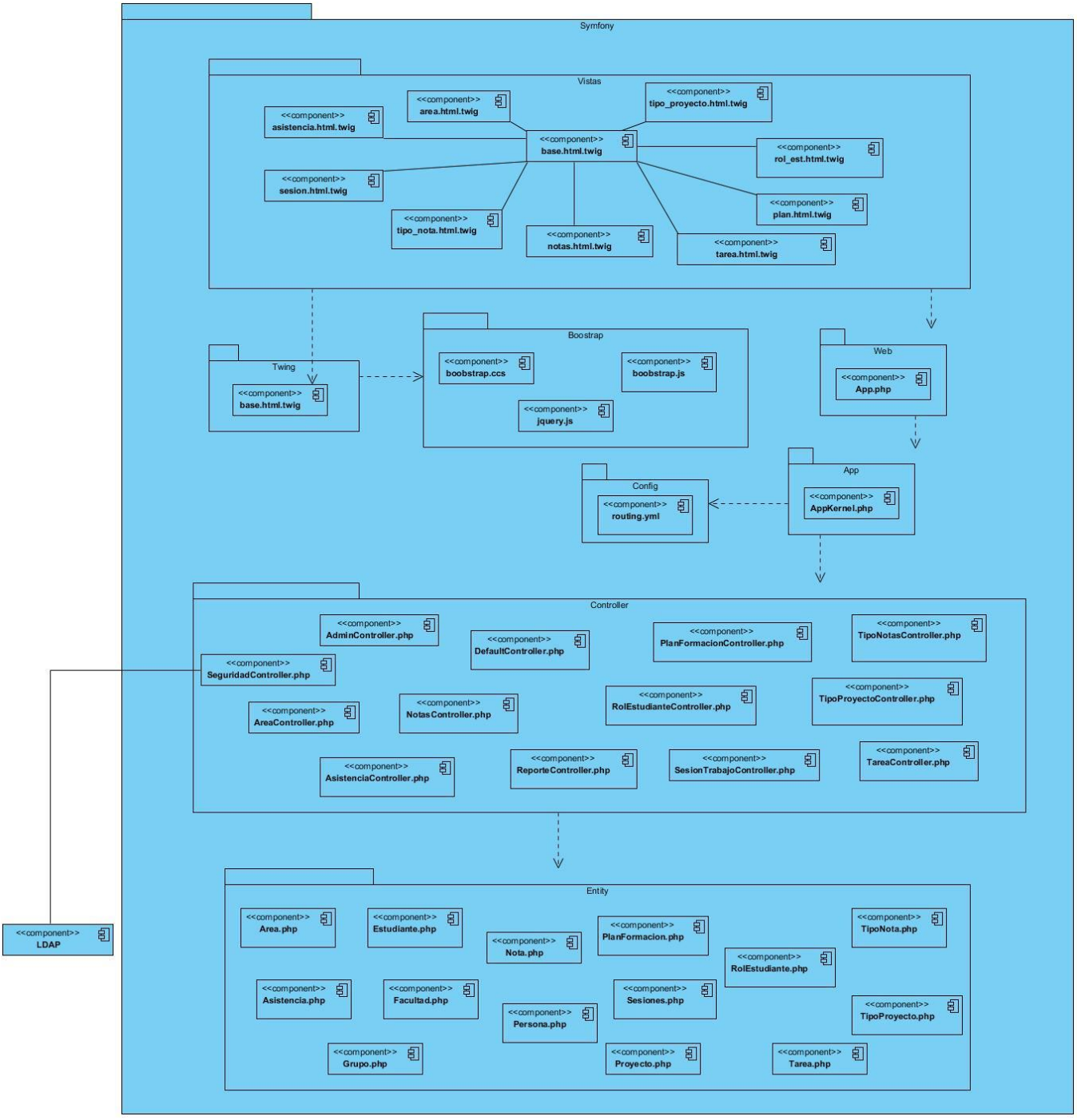


Figura 11. Diagrama de componentes. Elaboración propia

Los componentes contenidos en el diagrama anterior son:

- Vistas: representan las páginas clientes que permiten y facilitan la comunicación de los usuarios con el sistema.
- base.html.twig: es la plantilla base para la organización de las páginas html.twig.
- bootstrap.css: archivos de estilos de formato visual (CSS) de las páginas html.twig.
- bootstrap.js y jquery.js: archivos de validación de JavaScript (JS).
- App.php: componente que representa al controlador frontal o punto de acceso a la aplicación.
- AppKernel.php: representa a la clase responsable de cargar el archivo que gestiona las rutas de las clases que contienen las funcionalidades que responden a las peticiones del cliente y con las clases controladoras.
- routing.yml: archivo que contiene las rutas de las funcionalidades del sistema.
- Controller: clases controladoras que gestionan el flujo de información en el sistema.
- Entity: representan a las clases entidades que contienen la información persistente del sistema.
- LDAP: permite la autenticación y obtención de los datos de los usuarios del sistema, en este caso mediante un usuario y su contraseña del dominio UCI.

3.1.2 Interfaces de usuario

Una interfaz de usuario hace referencia a la interfaz con la que las personas interaccionan con las máquinas. Se trata de la interfaz que permite usar un ordenador para realizar un pedido en una tienda online o para acceder a una app a través de un *smartphone*. Esta cuenta con los elementos de control de una interfaz, que son visibles para el usuario o a través de los que puede efectuar alguna acción. En ello se incluyen desde simples líneas de comandos basadas en texto hasta interfaces gráficas de usuario con un diseño más complejo. La interfaz de usuario está íntimamente relacionada con la facilidad de uso de un software o de una página web. Hace tiempo que el objetivo ya no solo es conseguir una interfaz de usuario útil, sino que el aspecto estético también juega un papel importante. Por consiguiente, la interfaz de usuario es importante para una buena experiencia de usuario**,** es decir, para la experiencia que un usuario tiene groso modo con una página web o un software. (Digital Guide, Interfaz gráfica de usuario, 2017)

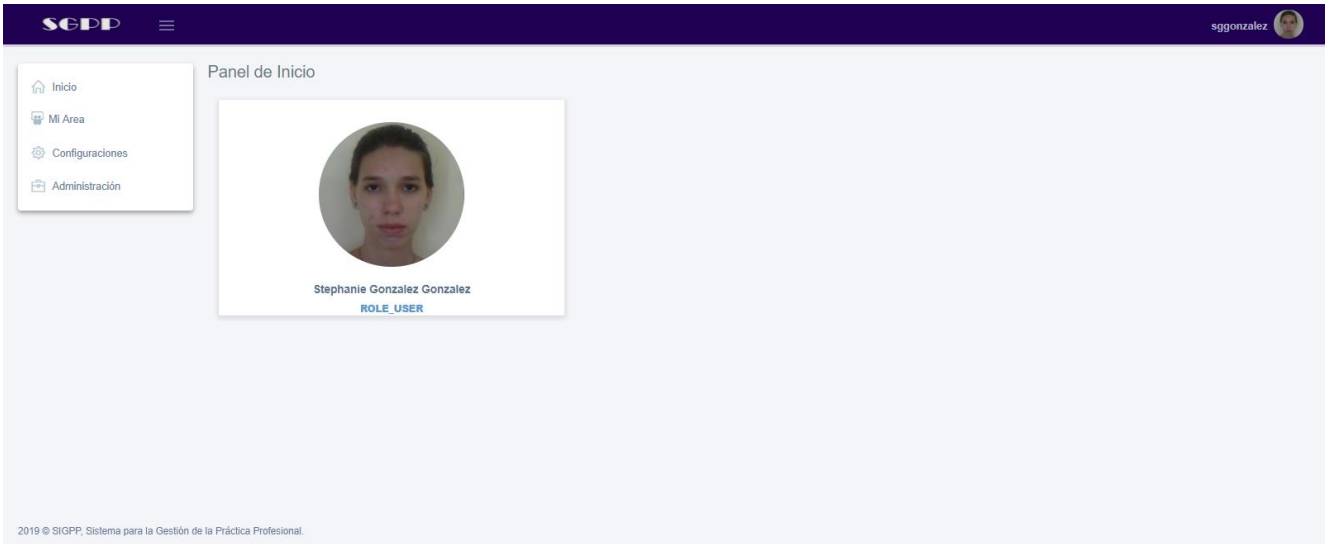


Figura 12. Interfaz de usuario Panel de inicio. Elaboración propia

3.2 Diagrama de despliegue

Un diagrama de despliegue muestra las relaciones físicas entre los componentes hardware y software en el sistema final, es decir la configuración de los elementos de procesamiento en tiempo de ejecución y los componentes software. Representan a los nodos y sus relaciones, los nodos son conectados por asociaciones de comunicación tales como enlaces de red, conexiones TCP/IP etc. (Guadarrama, Jorge;, 2013)

En la *Figura 13* se observa el diagrama de despliegue de la solución. En la computadora cliente se accede al sistema mediante el protocolo HTTPS. El sistema se encuentra instalado en un servidor web y maneja las peticiones que realiza el cliente. La información del sistema, de las acciones ejecutadas y del negocio es almacenada en la base de datos y mediante el protocolo HTTPS se realiza el consumo de los servicios necesarios para el correcto desarrollo del sistema, como el LDAP.

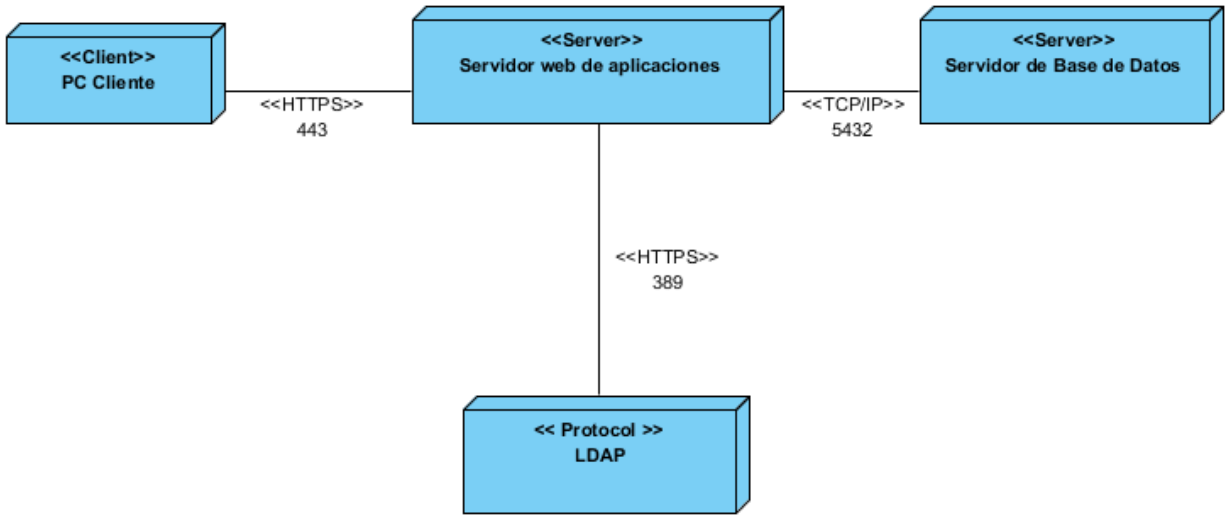


Figura 13. Diagrama de despliegue. Elaboración propia

3.3 Estándares de codificación

Durante el proceso de implementación de un software es considerado una buena práctica realizar la codificación del mismo siguiendo estándares que guíen este proceso. De tal manera que a los programadores se le facilita entender el código escrito por otros. Se realizan con el objetivo de facilitar el entendimiento y mantenimiento del código, su lectura y comprensión. El marco de trabajo Symfony3.4 emplea una estructura de código de la que a continuación se describen las empleadas en la solución. (Merkury, 2017) (Arturo Verbel de León, 2018)

Estructura del código

- Añadir un espacio después de cada operador binario (==, &&...) con la excepción del operador de concatenación (.).
- Añadir una coma después de cada *array* en un *array* multi-línea, incluso después del último.

- Añadir una línea en blanco antes de la sentencia *return*, a no ser que *return* esté sólo dentro de un grupo de sentencias (como en un *if*).
- Usar llaves {} para indicar la estructura de control, independientemente del número de declaraciones que contenga.
- Usar paréntesis para instanciar clases independientemente del número de argumentos que tiene el constructor.
- Declarar las propiedades de clase antes que métodos.

Nomenclaturas utilizadas

- Nomenclatura de los controladores: los controladores después del nombre llevan el sufijo Controller. Por ejemplo, AdminController.
- Nomenclatura de las variables: el nombre a emplear para las variables se escribe con la primera palabra en minúscula, en caso de que sea un nombre compuesto se empleará la notación *lowerCamelCasing* y comenzando con un prefijo según el tipo de datos. Por ejemplo $nombre y $planesFormacion.
- Uso de guiones bajos para nombres de opción y nombres de parámetros.
- Sufija las interfaces con *Interface*.
- Sufija las excepciones con *Exception*.
- Utiliza caracteres alfanuméricos y guiones bajos para los nombres de archivo.

3.4 Aplicación de las pruebas de software

La fase de pruebas del sistema tiene como objetivo verificar el sistema software para comprobar si este cumple sus requisitos. Dentro de esta fase pueden desarrollarse varios tipos distintos de pruebas en función de los objetivos de las mismas. Algunos tipos son pruebas funcionales, pruebas de usabilidad, pruebas de rendimiento, pruebas de seguridad y otras. (Guadarrama, Jorge;, 2013)

3.4.1 Pruebas de caja blanca

Son pruebas estructurales. Se obtienen casos de prueba que garanticen que se ejercitan por lo menos una vez todos los caminos independientes de cada módulo, ejerciten todas las decisiones lógicas en sus vertientes verdaderas y falsas, ejecuten todos los bucles en sus límites y con sus limitaciones

operacionales y ejerciten las estructuras internas de datos para asegurar su validez. (Carabali, Mauricio;, 2013)

Las técnicas usadas para ejecutar las pruebas se describen a continuación:

- **Prueba de flujo de datos:** por medio de esta herramienta se hace la selección más adecuada del flujo de datos, para llegar a una resolución correcta. Esto para probar las variables y definiciones en el programa.
- **Prueba de camino básico:** esta prueba demuestra el conjunto de pasos base del programa, lo que quiere lograr es que cada sentencia de código se ejecute mínimo una vez. Se usan herramientas como grafos, diagramas de flujo, la complejidad ciclomática entre otros.
- **Prueba de condición:** ejercita las condiciones lógicas contenidas en el módulo de un programa. Garantiza la ejecución por lo menos una vez de todos los caminos independientes de cada módulo, programa o método.
- **Prueba de bucles:** se centra en la validez de las construcciones de bucles. Garantiza la ejecución todos de todos los bucles en sus límites operacionales.

Con el objetivo de valorar la calidad de la implementación fue necesario aplicar una de las pruebas descritas con anterioridad: Prueba de camino básico. Para ello se siguieron los siguientes pasos:

- A partir del código fuente, dibujar el grafo de flujo asociado.
- Calcular la complejidad ciclomática del grafo.
- Determinar el conjunto básico de caminos independientes.
- Preparar los casos de prueba que obliguen a la ejecución de cada camino del conjunto básico.

Dando cumplimiento a los pasos básicos, se enumeran cada una de las sentencias de código de la funcionalidad *modificarArea()*, que es la encargada de modificar los datos del área seleccionada.

```php
public function editAction(Request $request, Area $obj)
{
    $editForm = $this->createForm('AppBundle\Form\AreaType', $obj);
    $editForm->handleRequest($request);

    if ($editForm->isSubmitted() && $editForm->isValid()) {
        if ($this->canPersist($obj)) {
            $this->getDoctrine()->getManager()->flush();
            $this->addFlash('success', 'Area editada con éxito');
            return $this->redirectToRoute('area_show', array('id' => $obj->getId()));
        }
    }

    return $this->render('area/edit.html.twig', array(
        'entity' => $obj,
        'edit_form' => $editForm->createView()
    ));
}
```

Figura 14. Fragmento del código del método modificarArea(). Elaboración propia

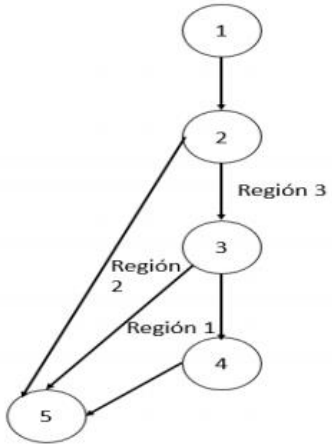


Figura 15. Grafo de flujo asociado al método modificarArea(). Elaboración propia

La complejidad ciclomática es una métrica inicialmente propuesta por Thomas J.McCabe en un artículo suyo en 1976 en el que escribía sobre la complejidad en el desarrollo de software. Esta métrica mide el número de flujos distintos de ejecución que puede tener el código de un artefacto de software, dicho llanamente, nos dice cuántos *ifs-then-else*, *while*, *for*, *switch*…etc., tenemos en nuestro código. Nos va a dar una medida cuantitativa de como de complejo va a ser comprender el código analizado. A más complejidad ciclomática, más complejo será el código, más complicado de leer, de entender, de modificar, de mantener y, por lo tanto, más caro. Además, obteniendo su valor, determina el número máximo de

pruebas que hay que realizar para asegurarnos que nuestra cobertura de test pasan al menos una vez por cada línea del código. (Pijierro, Manu;, 2018) El cálculo es necesario efectuarlo mediante tres vías o fórmulas de manera tal que quede justificado el resultado, siendo el mismo en cada caso:

Se calcula de la siguiente manera:

1. V (G)=E-N+2 donde E es el número de aristas y N los vértices.
   V (G)=6-5+2
   V (G)= 3
2. V (G) = P + 1 siendo “P” la cantidad de nodos predicados (son los nodos de los cuales parten dos o más aristas).
   V (G) = 2 + 1
   V (G) = 3
3. V (G) = R siendo “R” la cantidad total de regiones, se incluye el área exterior del grafo, contando como una región más
   V (G) = 3

El cálculo efectuado mediante las fórmulas antes presentadas muestra una complejidad ciclomática de valor tres, de manera que existen tres posibles caminos por donde el flujo puede circular, este valor representa el número mínimo de casos de pruebas para el procedimiento tratado.

Camino básico 1: 1-2-3-4-5.

Camino básico 2: 1-2-5.

Camino básico 3: 1-2-3-5.

Luego se definen los casos de prueba para comprobar la ejecución de cada camino del conjunto básico. En el diseño de los casos de prueba se deben especificar los siguientes elementos:

Descripción: contiene una descripción sobre las restricciones de los datos de entrada que debe tener el caso de prueba.

Condición de ejecución: se especifican los parámetros que debe poseer el caso de prueba para que se cumpla una condición deseada como respuesta del funcionamiento del procedimiento.

Entrada: se muestran los parámetros de entrada al procedimiento.

Resultados esperados: se explica el resultado esperado de la ejecución del procedimiento.

La tabla muestra el diseño de caso de prueba para el camino 1 del conjunto básico de caminos linealmente independientes, correspondiente a la funcionalidad modificarArea(). En este camino se prueba la modificación de los datos de un Área de forma satisfactoria.

Tabla 15. Diseño de caso de prueba para el camino 1. Elaboración propia

Diseño de caso de prueba para el camino 1	
Descripción	Los datos de entrada son válidos y deben cumplir con el formato especificado.
Condición de ejecución	Campos válidos: El campo Nombre, permite la entrada del nombre del área y es una cadena de caracteres no nula. El campo Es Grupo de Investigación es un campo de selección y puede ser nulo.
Entrada	Nombre: CISED Es Grupo de Investigación: No
Resultados esperados	Se modifican los datos del área satisfactoriamente.

Aplicándose el caso de prueba expuesto anteriormente se evidenció que el flujo de trabajo de las funcionalidades es correcto, ya que se comprobó que cada sentencia es ejecutada al menos una vez, cumpliéndose las condiciones de la prueba y el resultado esperado es satisfactorio.

3.4.2 Pruebas de caja negra

Son pruebas funcionales. Las pruebas se aplican sobre el sistema empleando un determinado conjunto de datos de entrada y observando las salidas que se producen para determinar si la función se está desempeñando correctamente por el sistema bajo prueba. Las herramientas básicas son observar la funcionalidad y contrastar con la especificación. El tipo de prueba a aplicar es la de Particiones de equivalencia. Este método intenta dividir el dominio de entrada de un programa en un número finito de clases de equivalencia. De tal modo que se pueda asumir razonablemente que una prueba realizada con un valor representativo de cada clase es equivalente a una prueba realizada con cualquier otro valor de dicha clase. Esto quiere decir que, si el caso de prueba correspondiente a una clase de equivalencia detecta un error, el resto de los casos de prueba de dicha clase de equivalencia deben detectar el mismo error y viceversa, si un caso de prueba no ha detectado ningún error, es de esperar que ninguno de los casos de prueba correspondientes a la misma clase de equivalencia encuentre ningún error. (Carabali, Mauricio;, 2013) El diseño de casos de prueba según esta técnica consta de dos pasos:

1. Identificar las clases de equivalencia.
2. Identificar los casos de prueba.

En el caso del Sistema para la gestión de la información de la PP en la Facultad, se diseñó un caso de prueba por cada requisito funcional. En la siguiente tabla se muestran los resultados de las pruebas realizadas.

Tabla 16. No conformidades detectadas por iteración. Elaboración propia

No conformidades	Aplicación	Ortografía
Primera iteración	20	10
Segunda iteración	8	3
Tercera iteración	-	-

3.5 Pruebas de Carga y Stress

Las pruebas de carga se realizan para determinar y validar la respuesta de la aplicación cuando es sometida a una carga de usuarios y/o transacciones que se espera en el ambiente de producción. Ejemplo: verificar la correcta respuesta de la aplicación ante el alta de 100 usuarios de forma simultánea. Se compara con el volumen esperado. Las pruebas de stress se realizan para encontrar el volumen de datos o de tiempo en que la aplicación comienza a fallar o es incapaz de responder a las peticiones. Superan los límites esperados en el ambiente de producción y/o determinados en las pruebas. Ejemplo: encontrar la cantidad de usuarios simultáneos, en que la aplicación deja de responder en forma correcta a todas las peticiones.
Todas las pruebas buscan encontrar cuellos de botella, de distinta manera. Se pueden realizan con las mismas herramientas, variando los parámetros indicados. (Juan, 2010)

Se realizaron las pruebas de carga y stress para casos críticos donde se encontraban conectados 150, 400 y 900 usuarios concurrentes. Las pruebas se desarrollaron con la ayuda de la herramienta Apache Jmeter, utilizando un ordenador con las siguientes características: Sistema operativo Windows 7, Microprocesador Intel Dual Core a 2.30 GHz, Memoria RAM 2GB. La propuesta de solución generó una buena transferencia de datos para 150 usuarios concurrentes esperados, lo que incurrió en un rendimiento de 1.5 seg, demostrando que la propuesta de solución es estable, ya que se mantuvo prestando servicios todo el tiempo sin incurrir en fallos. Para un total de 400 usuarios concurrentes el rendimiento fue de 2 seg con un porciento de error de 0.35 lo que significa que la propuesta sigue siendo bastante estable y para un total de 900 usuarios presentó un margen de error del 0.45% en un tiempo de respuesta de 2 seg lo que significa que la propuesta es lo suficientemente estable, no presentado problemas grabes en su estructura, ya que los problemas presentados fueron solamente en el desfasaje del tiempo de espera definido en los requisitos no funcionales. A continuación, se muestra una tabla con los resultados antes explicados.

Tabla 17. Resultados de aplicar prueba de carga y stress con el JMeter. Elaboración propia

Usuarios	Muestras	%Error	Tiempo de respuesta promedio
150	400	0.00%	1.5 seg

400	2100	0.35%	2 seg
900	4500	0.45%	2 seg

3.6 Aplicación y resultados de las pruebas de aceptación

En la Ingeniería del software, las pruebas de aceptación se realizan para establecer el grado de confianza en un sistema, partes del mismo o en sus características no funcionales. La confianza en el sistema estará determinada por su grado de adherencia a las necesidades, requerimientos y procesos de negocio solicitados por el usuario o cliente. Es en función a estos que el usuario debe decidir si acepta o no el sistema que le está siendo entregado. Por lo tanto, las pruebas de aceptación suelen ser responsabilidad de los clientes o usuarios del sistema. Otros interesados del proyecto pueden involucrarse también. (Pruebas de aceptación de software, 2016)

Pruebas Alfa: se lleva a cabo por un cliente, se usa el software de forma natural con el desarrollador como observador del usuario y registrando los errores y problemas de uso. Se llevan a cabo en un entorno controlado. (Software, pruebas alfa y beta, 2010)

Las pruebas se realizaron en un ambiente controlado utilizando un ordenador con las siguientes características: Sistema operativo Windows 10, Microprocesador Intel (R) Celeron (R) CPU 1005M a 1.90 GHz, Memoria RAM 4GB.

Las pruebas de aceptación se realizaron por parte de la Jefa de departamento de ISW de la Facultad1 y el Director del centro de desarrollo CISED. Para ello se realizaron las siguientes actividades:

- Revisión y aprobación de los requisitos funcionales y no funcionales por el cliente.
- Realización de pruebas exploratorias por parte del cliente al sistema desarrollado, para validar el cumplimiento de los requisitos aprobados.

Conclusiones del capítulo

En el capítulo se implementaron diagramas en los cuales se evidenció la dependencia de los componentes físicos y lógicos y a través de estos se obtuvo una visión más clara de la relación que existe

con la implementación de algunas clases del diseño de software y las relaciones entre ellas, además se mostraron protocolos para acceder a los diferentes servidores para el consumo de servicios. La interfaz de usuario hizo más visible el estado visual del Sistema para la gestión de la información de la Práctica Profesional para la aceptación por parte del cliente. Se utilizaron estándares de codificación para el mejor entendimiento y mantenimiento del código, lectura y comprensión, lo cual evidenció un avance más rápido preciso de la solución. Se realizó una fase de pruebas con el objetivo de verificar dicho sistema para comprobar si cumple con su objetivo general y los requisitos planteados, concluyendo en la realización de un Sistema para la gestión de la información de la Práctica Profesional con mayor calidad y mejores prestaciones.

CONCLUSIONES

- La fundamentación teórica y el análisis de los conceptos asociados a la investigación, permitió la comprensión de la realización del proceso de la práctica profesional y esclarecer así el proceso de gestión de la información académica.
- La sistematización del estado actual de las herramientas informáticas para el diseño de modelos entidad relación, y de la documentación analizada de sistemas homólogos, posibilitó la definición del ambiente de desarrollo para implementar el Sistema de gestión de la información de la Práctica Profesional.
- Integrar diversas áreas del conocimiento como la gestión y la ingeniería de software, programación y base de datos, permitió el análisis, diseño e implementación del Sistema para la gestión de la información de la Práctica Profesional en la Facultad dando cumplimento al objetivo modelación del diseño del sistema para la gestión de la información de la práctica profesional.
- La solución se validó a partir de la estrategia de pruebas definida, la que permitió obtener un Sistema para la gestión de la información de la práctica profesional con mejores prestaciones y calidad, a partir de los requisitos definidos por el cliente.
- La presente investigación concluyó con el desarrollo del Sistema para la gestión de la información de la práctica profesional, el cual contribuye a que el intercambio de información entre los centros y la facultad para el proceso de control y seguimiento de los estudiantes en la disciplina de Práctica Profesional sea en tiempo y sin pérdidas de información.

RECOMENDACIONES

- Realizar un módulo para guardar los proyectos realizados por los estudiantes en las prácticas.
- Distribuir el sistema por las áreas y facultades restantes de la Universidad.

REFERENCIAS BIBLIOGRAFICAS

Allamandri, Maxi. 2012. Prezi.com. *Modelo de diseño del software.* [En línea] 29 de 10 de 2012. [Citado el: 10 de 1 de 2019.] https://prezi.com/p4vuh3fapd5c/modelo-de-diseno-de-software/.

Almaraz Hernández, Jesús Matías, Campos Cantero, Pablo y Castelo Delgado, Tamara. 2011. *Desarrollo de una aplicación web para la gestión de Entornos Virtuales.* Madrid : s.n., 2011.

Alvares, Miguel Angel. 2014. desarrolloWeb.com. [En línea] 2 de 1 de 2014. [Citado el: 19 de 11 de 2018.] https://desarrolloweb.com/articulos/que-es-mvc.html..

Angel Luis Lozano Sánchez. 2016. ANGEL LOZANO. [En línea] 28 de 2 de 2016. [Citado el: 10 de 1 de 2019.] http://www.angellozano.com/requisitos-del-sistema-vs-casos-uso-vs-historias-usuario/.

Arenols Solano, Alex;. 2019. OpenWebinars. *Lenguajes de programación.* [En línea] 02 de 01 de 2019. [Citado el: 20 de 05 de 2019.] https://openwebinars.net/blog/12-caracteristicas-y-ventajas-de-php/.

Arquitectura de Software. **Cervantes, Humberto. 2018.** SG#27, Mexico : SG, 2018.

Arturo Verbel de León. 2018. micaminomaster.com.co. [En línea] 19 de 01 de 2018. [Citado el: 20 de 04 de 2019.] http://micaminomaster.com.co/herramientas-desarrollo/symfony-3-4-proyecto-backend/.

Bagarotti Abreu, Célida. 2017. *Sistema para la gestión de los procesos administrativos del CEIGE. Módulo para la Gestión de los procesos del Departamento de Práctica Profesional.* La Habana : s.n., 2017.

Carabali, Mauricio;. 2013. Prezi.com. [En línea] 26 de 09 de 2013. [Citado el: 11 de 03 de 2019.] https://prezi.com/sjwfwmix7slk/pruebas-de-caja-negra-y-caja-blanca/.

2017. Centers for medicare & medicaid services. [En línea] 2017. [Citado el: 19 de 11 de 2018.] https//www.cms.gov/Research-Statistics-Data-and-Systems/CMS-InformationTecnology/XLC/Downloads/SelectingDevelopmentApproach.pdf.

Chandra, V. 2015. International Journal of Computer Applications. [En línea] 2015. [Citado el: 19 de 11 de 2018.] http://pdfs.semanticscholar.org/e237/f9cb136f494c2bd0ce91525808c5c968b6b4.pdf.

Cillero, Manuel. 2016. manuel.cillero.es. [En línea] 2016. [Citado el: 12 de 12 de 2018.] https://manuel.cillero.es/doc/metrica-3/tecnicas/diagrama-de-clases/.

de la Torre Matamoro, Juan Carlos. 2013. *Web docente para la Práctica Profesional I de la Ingeniería Informática, en la semipresencialidad.* La Habana : s.n., 2013.

Delgado, Hugo. 2018. akus.net. *Blog Diseño Web.* [En línea] 24 de 10 de 2018. [Citado el: 19 de 11 de 2018.] https://disenowebakus.net/conozca-a-php.php.

REFERENCIAS BIBLIOGRAFICAS

—. 2018. akus.net. *Blog Diseño Web.* [En línea] 24 de 10 de 2018. [Citado el: 19 de 11 de 2018.] https://disenowebakus.net/javascript.php.

Desarrollo Web; PHP7: más velocidad de carga y acceso en Internet. 2018. Digital Guide. [En línea] 08 de 11 de 2018. [Citado el: 24 de 04 de 2019.] https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/caracteristicas-y-ventajas-de-php7/.

Digital Guide, Interfaz gráfica de usuario. 2017. Digital Guide. *Diseño Web.* [En línea] 29 de 08 de 2017. [Citado el: 20 de 03 de 2019.] https://www.ionos.mx/digitalguide/paginas-web/diseno-web/ui-que-es-una-interfaz-de-usuario/.

Editorial Definición MX. 2014. Definición MX. *Práctica Profesional.* [En línea] 06 de 07 de 2014. https://definicion.mx/practica-profesional/.

Eguiluz, Javier. 2012. Desarrollo web ágil con Symfony2. *Desarrollo web ágil con Symfony2.* 2012.

Gestion de información. 2017. instituciones.sld.cu. [En línea] 16 de 04 de 2017. [Citado el: 15 de 10 de 2018.] https://instituciones.sld.cu/toximed/2017/04/16/que-es-gestion-de-la-informacion/.

Guadarrama, Jorge;. 2013. Prezi.com. [En línea] 25 de 11 de 2013. [Citado el: 2019 de 02 de 15.] https://prezi.com/ijpanrlnvhyj/modelo-de-implementacion/.

Guerra, César Arturo. 2016. Obtención de Requerimientos, técnicas y estrategia. *SG.* [En línea] 2016. [Citado el: 4 de 12 de 2018.] https://sg.com.mx/revista/17/obtencion-requerimientos-tecnicas-y-estrategia.

Huanca Velarde, Juan Jose. 2014. Academia.edu. [En línea] 2014. [Citado el: 19 de 11 de 2018.] http://www.academia.edu/6045417/Visual_Paradigm..

Juan. 2010. Blog. [En línea] 25 de 03 de 2010. [Citado el: 20 de 04 de 2019.] http://juanbevilacqua.blogspot.com/2010/03/pruebas-de-estres-carga-y-rendimiento.html.

Leiva, Antonio. 2016. devexperto.com. *Patrones de diseño de software.* [En línea] 18 de 05 de 2016. [Citado el: 20 de 03 de 2019.] https://devexperto.com/patrones-de-diseno-software/.

MADEJA. 2015. Marco de desarrollo de la junta de Andalucía. [En línea] MADEJA, 2015. [Citado el: 10 de 1 de 2019.] http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/419.

MDN webs docs, CCS3. 2016. Developer.mozilla.org. [En línea] 2016. [Citado el: 19 de 11 de 2018.] https://developer.mozilla.org/es/docs/Web/CSS/CSS3.

Merkury. 2017. www.ohmyroot.com. *Estandares de codificacion.* [En línea] 12 de 01 de 2017. [Citado el: 15 de 04 de 2019.] https://www.ohmyroot.com/buenas-practicas-legibilidad-del-codigo/.

Mifsuf Talón, Elvira. 2012. Apache. [aut. libro] Elvira Mifsuf Talón. *Apache.* España : Ministerio de Educación, Cultura y Deporte- Área de Educación, 2012.

REFERENCIAS BIBLIOGRAFICAS

Ministerio de Educación Superior. 2014. *Plan de estudio "D" Ingenieria en Ciencias Informáticas.* La Habana : s.n., 2014.

—. 2014. *Plan de Estudios ¨D¨ Ingeniería en Ciencias Informáticas.* La Habana : Ministerio de Educación Superior, 2014.

Muñoz García, Adolfo , Adelantado Mateu, Eulalia y Canet Centellas, Fernando. 2002. *Aplicación web para la gestión del conocimiento en las asignaturas de producción y realización de audiovisuales.* Valencia : s.n., 2002.

Núñez Mayorga, Gloria María. 2009. *Desarrollo de una aplicación web para la gestión de calificaciones de alumnos.* Madrid : s.n., 2009.

Object Management Group. 2005. Unified Modeling Language UML. [En línea] 2005. [Citado el: 19 de 11 de 2018.] http://www.uml.org.

Pérez Porto, Julián. 2015. Definición.de. [En línea] 2015. http://definicion.de/practica-profesional/.

Pijierro, Manu;. 2018. A Medium Corporation. [En línea] 27 de 09 de 2018. [Citado el: 25 de 03 de 2019.] https://medium.com/@mpijierro/complejidad-ciclom%C3%A1tica-y-como-reducirla-7374c215f666.

PostgreSQL 9.4. 2014. www.postgresql.org. [En línea] 18 de 12 de 2014. [Citado el: 15 de 10 de 2018.] https://www.postgresql.org/about/press/presskit94/es/.

Pruebas de aceptación de software. 2016. PMOinformatica.com. [En línea] 08 de 08 de 2016. [Citado el: 01 de 05 de 2019.] http://www.pmoinformatica.com/2016/08/pruebas-aceptacion-software-istqb.html.

Quiroga, Juan Pablo. 2012. *Requerimientos Funcionales y No Funcionales.* Universidad de los Andes : s.n., 2012.

Reinoso Miranda, and Aguiar Godoy, Yoan Manuel. 2015. *Sistema de gestión de información para el departamento de Práctica Profesional del Centro FORTES.* La Habana : s.n., 2015.

Reyes R., Leidy, Ruiz, Maria del Carmen y Vivanco E., Mónica. 2009. Ingenieria en Sistemas 2009 UNL. *Prototipos Informaticos.* [En línea] 2009. [Citado el: 10 de 1 de 2019.] https://sistemas2009unl.wordpress.com/prototipos-informaticos/.

Rodríguez Sánchez, Tamara. 2014. *Metodología de desarrollo para la Actividad Productiva de la UCI.* La Habana : s.n., 2014.

Romero, Hemeregildo. 2009. SlideShare. *Metodologías de desarrollo.* [En línea] 2009. [Citado el: 19 de 11 de 2018.] http://es.slideshare.net/MeneRomero/metodologias-de-desarrollo.

Sarboraria, Matthew;. 2018. Oracle Corporation. [En línea] 2018. [Citado el: 6 de 12 de 2018.] https://www.oracle.com/co/mysql/.

Sistema de Gestión de Información para la Educación Superior. **Cano Iclán, Anisley, Campillo Torres, Irima y Cuesta Rodríguez, Floriselda. 2014.** 2, 2014, Vol. 46.

REFERENCIAS BIBLIOGRAFICAS

Software, pruebas alfa y beta. 2010. WordPress.com. [En línea] 08 de 09 de 2010. [Citado el: 01 de 05 de 2019.] https://rfolivares.wordpress.com/2010/09/08/pruebas-alfa-y-beta/.

Symfony. 2015. Symfony.es. [En línea] 2015. [Citado el: 19 de 11 de 2018.] https://symfony.es.

UCI. 2013. uci.cu. *investigación y desarrollo.* [En línea] 2013. [Citado el: 19 de 11 de 2018.] https://www.uci.cu/investigacion-y-desarrollo/productos/xedro/gespro-1305.

Uninotas. 2016. Uninotas. [En línea] 21 de 12 de 2016. [Citado el: 10 de 1 de 2019.] https://www.uninotas.net/revisiones-tecnicas-formales/.

Universidad Nacional abierta y a distancia. *Lenguaje de Modelado Unificado UML.* [En línea]

ANEXOS

Anexo1: Centros productivos, Líneas de investigación, Dirección de investigación y otras áreas donde se realiza la práctica profesional de los estudiantes. (Existentes hasta la fecha)

Centros Productivos
Centro de Ideoinformática (CIDI)
Centro de Software Libre (CESOL)
Centro de Identificación y Seguridad Digital (CISED)
Centro de Telemática (TLM)
Centro de Información de la Gestión Documental (CIGED)
Centro de Información de la Gestión a Entidades (CEIGE)
Centro de Gobierno Electrónico (CEGEL)
Centro de Tecnología para la Formación (FORTES)
Centro de Tecnologías y Desarrollo de Arquitecturas Empresariales (CDAE)
Centro de Informática Industrial (CEDIN)
Centros de Entornos Interactivos 3D (VERTEX)
Centro de Geoinformática y Señales Digitales (GEYSED)
Centro de Tecnologías y Gestión de Datos (DATEC)
Centro de Informática Médica (CESIM)
Centro de SOPORTE

Líneas de Investigación, dirección de investigación y otras áreas
Dirección de Informatización (DIN)
Empresas de Tecnologías de la Información para la Defensa (XETID)
Línea de Inteligencia Artificial y Reconocimiento de Patrones (LIARP)
Centro de Educación a Distancia (CENED)
Dirección de CALIDAD
Línea de Informática Aplicada (LIA)
UCIMININT
Dirección de Investigación (DIRINV)
Dirección de Seguridad Informática (DRSI)
Centro de Matemática Computacional (CEMC)

Anexo2: Entrevista de expertos vinculados a la práctica profesional.
Nombre del entrevistado(a): Aylin Estrada Velazco, Vicedecana de Formación de la Facultad 1.
Preguntas:
¿Cómo se realiza el flujo de la práctica profesional?
¿Cuántos estudiantes se encuentran dando la Práctica Profesional actualmente en la facultad?
¿Cuántos estudiantes se encuentran dando la Práctica Profesional en otras áreas de la universidad?
¿Cuántas áreas existen en la universidad impartiendo la práctica profesional?
¿Qué documentación se genera?
¿Cómo se procede a la evaluación d la misma?

ANEXOS

Anexo2: Entrevista para el levantamiento de requisitos.
Nombre del entrevistado(a): Osay González Fuentes, Director del Centro de Identificación y Seguridad Digital y Mayleidis López Fernández, Jefa de departamento de ISW Facultad 1.
Preguntas:
¿Cuál es el objetivo del sistema a desarrollar?
¿De cuántos módulos debe constar el sistema que se desea desarrollar?
¿Qué información académica se debe gestionar?
¿Debe generar reportes a los profesores?
¿Qué propiedades debe tener el servidor donde se deberá correr el sistema?