



Universidad de las Ciencias Informáticas

*Trabajo de diploma para optar por el título de
Ingeniero en Ciencias Informáticas*

***“Registro de información para la caracterización de los estudiantes
de la Universidad de las Ciencias Informáticas”***

Autores:

Dreidys R. Reyes Cueto

Dany Ferrera Ches

Tutores:

Ing. Leyany Yera Moya

Ing. Aylenn Aquino Leiva

Ing. Lianet Liben Martínez

La Habana, junio de 2012

“Año 54 de la Revolución”



“...aquí está una de las tareas de la juventud: empujar y dirigir con el ejemplo la producción del hombre del mañana...”

de

Declaración de autoría

Declaramos ser autores del presente trabajo de diploma y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo. Autorizamos a dicho centro para que haga el uso que estime pertinente con este trabajo.

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

Dreidys R. Reyes Cueto

Firma del autor

Dany Ferrera Ches

Firma del autor

Ing. Leyany Yera Moya

Firma del tutor

Ing. Aylienn Aquino Leiva

Firma del tutor

Ing. Lianet Liben Martínez

Firma del tutor

Agradecimientos

Dreidys

Agradezco a mis padres por haberme apoyado siempre en las buenas y malas, a mi hermana, mis sobrinos y mi cuñado por estar cuando los necesitaba y ayudar a que este sueño se hiciera realidad. A mi novio por brindarme con amor su apoyo y compañía en los buenos y malos momentos contribuyendo a obtener este logro tan especial en mi vida. Le doy gracias a Adrian Torres por haberme soportado durante estos 5 años y haberme ayudado siempre que lo necesité incondicionalmente.

A mis 2 amigas Mary Kry y Laura.

A Nardo por ser para mí, como un padre.

A mis compañeros del aula y profesores.

A las muchachitas del apratamento la Mía (Maryanis), Ariamnis, Gruñona (Yudith), Yasne, Zoe y a mis compañeros del proyecto (Súper leo, Iván, Yaniel, Rodain, Enrique, Alianny, Soima).

Gracias a todas aquellas personas que de una forma u otra aportaron su granito de arena y confiaron en mí.

Dreidys y Dany

Agradecemos a nuestras tutoras, que de una forma u otra siempre estuvieron al tanto y muy preocupadas para que se lograra este sueño tan grande. Muchas gracias por todo.

Dany

Agradecimientos

Quisiera agradecer a todas las personas que de una forma u otra se han hecho parte de este sueño hecho realidad:

A mi madre (Ana Caridad Ches Viera) y a mi tía (Oda):

Por estar siempre a mi lado, por ayudarme a levantarme ante cada caída y enseñarme que en la vida siempre hay una salida, por confiar en mí y por siempre estar a mi lado.

A mi hermano (Norberto Javier Ferrera Ches):

Por quererme tanto y ayudarme cada vez que lo necesité.

A mi abuela (Mariana):

Por ser la mejor abuela del mundo. Gracias por todo.

A mi familia en general que siempre me han apoyado tanto y que nunca dudaron de mí:

A mis abuelos queridos.

A mis tías y tíos.

A mis primas y primos.

A todos mis amigos y compañeros de grupo.

A todos mis profesores desde 1er año hasta 5to año.

Al Combo FC (Miche, Migue, Ernesto, LachyMen, El Trujo, Alexis, Omar, German, Mulato, Frank, El Flecha, Mariño):

Por esos momentos de felicidad que pasamos todos juntos.

Dreidy

A mi mamita por confiar en mi y darme ánimo cada vez que lo necesité, por vivir conmigo la aungustiosa espera de cada resultado, por sus consejos que me han hecho salir adelante en la vida y por brindarme ese amor incondicional que solo una madre sabe dar. A mi hermana y mis sobrinos que siempre hacen tuyas mis alegrías y tristezas y aunque no se encuentren en este momento se que están orgullosos de mi, y a mi papá por ser una de las personas que mas quiero en al vida. Los amo

Dany

A mis padres y a mi hermano por enseñarme a dar mis primeros pasos en la vida. Ya era hora de recoger la cosecha sembrada hace 23 años atrás. Los quiero.

Resumen

Actualmente en la Universidad de las Ciencias Informáticas (UCI) el proceso de registro de información para las caracterizaciones de los estudiantes resulta no confiable, debido a que el sistema que se tiene actualmente en uso realiza dicho proceso en el último año de la carrera por medio de los tutores, los cuales deben registrar todas las actividades realizadas por el estudiante en el transcurso de su carrera sin tener la posibilidad de comprobar de que toda la información registrada es verídica. Se plantea un sistema de registro de información para las caracterizaciones de los estudiantes en cualquier momento de su vida universitaria, para ello se hace necesario un registro que mantenga un seguimiento real de toda la información asociada a los mismos en las diferentes esferas que se evalúan en la universidad, dígase formación, extensión e investigación-producción, además de tener en cuenta su trayectoria FEU-UJC, medidas disciplinarias y las competencias técnicas adquiridas. Lograr este objetivo en una universidad poblada por un gran número de estudiantes es una tarea difícil si no se utilizan las Tecnologías de la Información y las Comunicaciones (TIC) como soporte para su aplicación. En este trabajo se presenta la conceptualización e implementación de este sistema mediante el uso de herramientas de *software* libre.

Palabras clave: caracterización de estudiantes, gestión de información de la trayectoria estudiantil, registro de información de la trayectoria estudiantil

Índice

Introducción	1
Capítulo 1: Fundamentación teórica	5
1.1 Introducción.....	5
1.2 Conceptos asociados al dominio del problema.....	5
1.3 Análisis de sistemas homólogos.....	6
1.4 Proceso de desarrollo ágil con segundo nivel de CMMI (<i>Capability Maturity Model Integration</i>).....	8
1.5 Lenguaje de modelado	11
1.6 Herramientas y tecnologías	12
1.6.1 Visual Paradigm 8.0.....	12
1.6.2 Evolus Pencil 1.3	13
1.6.3 Apache JMeter 2.6.....	13
1.6.4 Marco de desarrollo	14
1.6.5 Sistema gestor de base de datos.....	14
1.6.6 Servidor web.....	15
1.6.7 Entorno integrado de desarrollo «IDE»	16
1.6.8 Lenguajes de programación.....	16
1.7 Conclusiones parciales.....	17
Capítulo 2: Análisis y diseño del sistema	19
2.1 Introducción.....	19
2.2 Modelo del negocio	19
2.3 Propuesta de solución	21
2.4 Definición de los requisitos funcionales y no funcionales.....	22
2.4.1 Requisitos funcionales	22
2.4.2 Definición de los requisitos funcionales.....	22

2.4.3 Especificación de los requisitos funcionales.....	26
2.5 Requisitos no funcionales.....	30
2.6 Técnicas de obtención de requisitos.....	33
2.7 Modelo físico de la Base de Datos	34
2.8 Estilo arquitectónico	35
2.9 Patrón de arquitectura	36
2.10 Patrones de diseño.....	37
2.11 Aplicación de los patrones.....	39
2.12 Modelo de despliegue.....	40
2.13 Distribución de las funcionalidades del módulo	41
2.14 Integración de la propuesta de solución al Sistema de Gestión Académica de Pregrado.	43
2.14.1 Seguridad	43
2.14.2 Configuración.....	43
2.14.1 Trazas.....	43
2.14.2 Personal y Secretaría.....	43
2.15 Conclusiones parciales.....	44
Capítulo 3: Implementación y prueba.....	45
3.1 Introducción.....	45
3.2 Estándar de codificación.....	45
3.2.1 Denotación, llaves de apertura, cierre y tamaño de líneas	45
3.2.2 Convención de nomenclatura.....	46
3.2.3 Estructuras de control	48
3.2.4 Documentación	50
3.2.5 Buenas prácticas	51
3.3. Validación de los Requisitos Funcionales.....	52

Índice de contenido

3.3.1 Técnicas de validación.....	52
3.4 Pruebas.....	52
3.4.1 Niveles de pruebas	52
3.4.2 Métodos y técnicas de prueba	53
3.5 Conclusiones parciales.....	61
Conclusiones	62
Recomendaciones	63

FIGURA 1: FASES DEL PROCESO DE DEASRROLLO DE SOFTWARE	8
FIGURA 2: NIVELES DEL CMMI.....	10
FIGURA 3: MODELO DE DOMINIO.....	21
FIGURA 4: MODELO FÍSICO DE LA BASE DE DATOS	35
FIGURA 5: DIAGRAMA DE DESPLIEGUE	40
FIGURA 6: MAPA DE NAVEGACIÓN.....	42
FIGURA 7: TÉCNICAS DE CAJA NEGRA Y CAJA BLANCA.....	54
FIGURA 8: CÓDIGO DE REGISTRAR ACTIVIDADES DE EXTENSIÓN	55
FIGURA 9: GRAFO DE LOS CAMINOS DEL MÉTODO.....	56
FIGURA 10: NO CONFORMIDADES.....	60
FIGURA 11: RESULTADOS DE PRUEBAS DEL JMETER	61

Índice de tablas

TABLA 1: REQUISITOS FUNCIONALES	26
TABLA 2: BUSCAR ESTUDIANTE	28
TABLA 3: CREAR REGISTRO DE INFORMACIÓN DE FORMACIÓN	30
TABLA 4: REQUISITOS NO FUNCIONALES	33
TABLA 5: MOSTRAR DETALLES DE METODOLOGÍAS.....	59

Introducción

Son muchas las instituciones y empresas que han comenzado a llevar a cabo el proceso de informatización de su actividad, lo que provee una mejora en el desarrollo de los procesos al permitir que se almacenen grandes volúmenes de información y a su vez, facilitar la búsqueda adecuada de la misma por parte de científicos, investigadores, profesionales, estudiantes y trabajadores. En el país se lleva a cabo el proceso de informatización de la sociedad, por lo que hoy día no se puede hablar de educación, salud, cultura, progreso sino se aplican las TIC. Actualmente las empresas cubanas afrontan un cambio revolucionario de pensamiento acerca de cómo el uso de los sistemas informáticos puede sin duda alguna agilizar las actividades que se realizan a diario.

La UCI, concebida para contribuir al desarrollo de la informática en Cuba, ha desempeñado un papel fundamental en el desarrollo de *software* no solo para entidades nacionales, sino también internacionales. Actualmente existen diversos centros productivos cuya misión fundamental es el desarrollo de *software*. El Centro de Informatización Universitaria (CENIA) se encarga de informatizar e integrar los procesos sustantivos de la universidad. Existe además el Centro de Innovación y Calidad de la Educación (CICE) encargado de contribuir a elevar la calidad del proceso de formación integral de los estudiantes en la UCI.

El CICE utiliza el Sistema de Caracterización Integral del Graduado (SCIG) con el objetivo de registrar a los estudiantes de quinto año su trayectoria en la universidad en las distintas esferas evaluadas: formación, extensión, investigación-producción, además de incluir su trayectoria FEU-UJC; detectando en cada una de estas esferas sus capacidades y potencialidades. La información de este sistema es registrada por el tutor sin tener fuentes confiables que avalen que lo que se está registrando es correcto, por lo que no se puede tener un seguimiento real de las actividades realizadas por los estudiantes a lo largo de la carrera. La información que obtiene el SCIG no se encuentra centralizada, puesto que es adquirida de diferentes sistemas mediante el uso de servicios web, lo que trae consigo que estos servicios no sean los más óptimos debido a su lentitud a la hora de acceder a la información.

El Sistema de Gestión Universitaria (SGU) se encarga de desarrollar el subsistema base para la gestión de los procesos universitarios, con el objetivo de proveer un sistema integrado que informatice los procesos de la comunidad universitaria referente a gestión de pregrado, posgrado, residencia, investigaciones y cooperación internacional [1]. El CICE como proceso de la comunidad universitaria es necesario integrarlo a dicho sistema obteniendo los beneficios y facilidades provistas por dicha

integración, entre ellas consultar toda la información de manera centralizada, así como los beneficios del uso de la arquitectura de *software* utilizada en dicho sistema. Que no se cuente con una aplicación capaz de registrar la información requerida de los estudiantes en los diferentes momentos de su carrera, así como no integrarse al SGU trae como consecuencia que el mismo se realice de forma no confiable y engorrosa.

Teniendo en cuenta lo anteriormente planteado el **problema a resolver** queda formulado por la siguiente interrogante: ¿Cómo registrar la información de la trayectoria estudiantil en la Universidad de las Ciencias Informáticas?

Se define como **objeto de estudio**: proceso de registro de información enmarcando su **campo de acción** en la gestión del proceso de registro de información de la trayectoria estudiantil en la Universidad de las Ciencias Informáticas.

La presente investigación tiene como **objetivo general**: desarrollar un sistema informático que permita con su integración al Sistema de Gestión Universitaria un mejor desarrollo del proceso de registro de la información para la caracterización de los estudiantes de la UCI.

Para darle cumplimiento al objetivo planteado se proponen las siguientes **tareas de investigación**:

- Definición de conceptos relacionados con la información para la caracterización de estudiantes y su registro.
- Estudio de los principales sistemas de registro de información estudiantil existentes para conocer sus características y funcionalidades, así como sus ventajas.
- Análisis y diseño del sistema de registro de información para la caracterización de los estudiantes de la UCI.
- Implementación de los requisitos obtenidos en el análisis.
- Validación de las funcionalidades del sistema.

Con el propósito de lograr una mayor organización en la realización de la presente investigación se recurre a los siguientes métodos:

Métodos teóricos

Dentro de los métodos teóricos utilizados se encuentran:

Histórico-lógico: este método es utilizado para realizar el estudio del estado del arte con respecto al registro de información estudiantil, para de esta forma lograr un análisis eficiente capaz de determinar su situación actual y de definir en el plano teórico los principales elementos de la investigación.

Analítico-sintético: este método es utilizado para analizar teorías y documentos acerca del registro de información, con el objetivo de facilitar la comprensión de la información recopilada y arribar a conclusiones válidas.

Modelación: este método es utilizado para la representación abreviada de la situación real del negocio, a través de modelos o diagramas que permitan representar los procesos del negocio.

Dentro de los métodos empíricos, se utilizaron:

Entrevista: para conocer las causas y que soluciones se le dan actualmente a los problemas sobre el registro de información para la caracterización de los estudiantes y si existen normas que rigen dicho registro.

Aportes prácticos esperados del trabajo

Disponer de un sistema informático que permita el registro de información para la caracterización de los estudiantes de la Universidad de las Ciencias Informáticas de manera organizada, obteniendo los siguientes beneficios:

- Realizar el registro de información desde que el estudiante ingrese a la universidad hasta que concluya sus estudios superiores.
- Realizar el registro de información en cualquier momento del curso académico.
- Consultar la trayectoria estudiantil facilitándole a los profesores llevar el seguimiento de los estudiantes para la toma de decisiones.
- Permitirá suministrarle información al Módulo de Caracterización del sistema CICE con el objetivo de realizarles a los estudiantes el proceso de caracterización.

La presente investigación está compuesta por tres capítulos, estructurados de la siguiente manera:

Capítulo 1. Fundamentación teórica: en este capítulo se presentan los elementos teóricos que sirven de base a la investigación del problema planteado, analizando los principales conceptos relacionados con el objeto de estudio. Se realiza un análisis de otras soluciones existentes y se presentan las

herramientas y metodología a utilizar para el desarrollo del sistema propuesto.

Capítulo 2. Análisis y diseño del sistema: en este capítulo se describe el análisis y diseño de la solución. Se realiza la descripción de los procesos de negocio, determinando las personas que interactuarán con el sistema. Se explica el flujo actual de los procesos y el objeto de informatización, se describen los requisitos realizando una propuesta del sistema y cómo será el flujo de los procesos en el mismo. Se realiza la descripción de las principales funcionalidades del sistema y la integración de este al Sistema de Gestión Universitaria. Se especifican además los patrones de diseño y los elementos de la arquitectura.

Capítulo 3. Implementación y prueba: en este capítulo se describen las técnicas y estándares de codificación, así como el tratamiento de errores. Se describen la ejecución de diferentes tipos de prueba para verificar que el sistema cumpla con los requerimientos deseados y los resultados de las mismas.

Además de conclusiones, recomendaciones, referencias bibliográficas, bibliografía consultada y anexos.

Capítulo 1: Fundamentación teórica

Capítulo 1: Fundamentación teórica

1.1 Introducción

El presente capítulo tiene como objetivo abordar los elementos teóricos que en su conjunto constituyen la base para la fundamentación teórica de la presente investigación. Se analizan los conceptos fundamentales relacionados con la gestión de información y caracterización de estudiantes. También se realiza un análisis sobre los sistemas que gestionan la información de estudiantes. Además, se hace un estudio sobre las principales características y ventajas de las herramientas, lenguajes y tecnologías a utilizar en el desarrollo de la investigación.

1.2 Conceptos asociados al dominio del problema

Caracterización de estudiantes: es el resultado que se obtiene después de caracterizar a una persona, determinando las cualidades o rasgos característicos de la misma [2]. En este caso se les aplica a los estudiantes con el objetivo de medir su integralidad con respecto a todas las actividades y eventos que se realizan en la universidad. Esta caracterización es de gran importancia ya que permite tener conocimiento del desenvolvimiento del estudiante, convirtiéndose la misma como en un currículo estudiantil que les servirá para su ubicación laboral al llegar a quinto año.

Gestión de información: la gestión de información es el proceso que se encarga de suministrar los recursos necesarios para la toma de decisiones, así como para mejorar los procesos, productos y servicios de las organizaciones [3]. Es el proceso mediante el cual se obtiene, despliega o utiliza una variedad de recursos básicos para apoyar los objetivos de la organización. Su utilidad está dada en función de su aporte a los procesos de toma de decisiones, creación de productos y solución de problema.

Información: es la acción de informar o informarse, averiguación de un hecho, dato o noticia. Agregación de datos que tiene un significado específico más allá de cada uno de estos. Un ejemplo: 2, 0, 0 y 1 son datos; 2001 es una información. La información ha sido siempre un recurso muy valioso, revalorizado hoy más aún por el desarrollo y la expansión de las TIC [4]. Permite resolver problemas y tomar decisiones. En el trabajo la misma está constituida por un grupo de datos ya supervisados y ordenados que sirven para contribuir a la caracterización de los estudiantes de la UCI.

Registro de información de la trayectoria estudiantil: es la acción de recolectar todos los

Capítulo 1: Fundamentación teórica

indicadores que sean capaces de evaluar al estudiante con respecto a su trayectoria en la universidad en las diferentes áreas. Permitiendo el acceso a la información siempre que se desee consultarla.

Sistema de gestión de información: los sistemas de gestión de la información son sistemas de procesamiento de datos que son diseñados para suministrar la información necesaria a la administración para tomar decisiones. Se puede añadir además que estos son utilizados para reunir los recursos financieros, la producción, y demás información que los administradores necesitan para operar un negocio, especialmente un sistema que es computarizado [5].

1.3 Análisis de sistemas homólogos

A continuación se abordan sistemas nacionales e internacionales para el registro de información con el fin de conocer si pueden ser adaptados y utilizados como solución a la problemática planteada. Además es objetivo de este estudio analizar el funcionamiento y características generales de dichos sistemas para tomarlos en cuenta en la solución que se propone.

Sistema de gestión escolar en Quito Ecuador

El sistema de gestión escolar ayuda con la gestión académica, administrativa y de comunicación a instituciones educativas. Los clientes que adquieren este sistema tienen acceso desde la institución, casas u oficinas a través de internet. Es una herramienta muy útil para el manejo de centros educativos tanto en el aspecto académico como administrativo, brinda servicios a profesores, personal administrador, padres de familia y estudiantes permitiendo llevar un mejor control sobre lo que sucede en la institución y quedando todo registrado [6].

Este sistema realiza el registro de información de forma organizada por cada uno de los aspectos que responden a sus necesidades, lo cual brinda al trabajo la posibilidad de guiar el registro enfocado a cada una de las esferas que se tratan en la universidad. Este no se utiliza ya que registra información que solo se adapta a sus necesidades y el registro se realiza a través de internet.

Sistema de Información Estudiantil (SIE)

El Sistema de Información Estudiantil, es la iniciativa tecnológica más ambiciosa del Departamento de Educación de Perú. Este sistema comprende un amplio repositorio de datos que almacena y maneja toda la información relacionada a los estudiantes y procesos de enseñanza en el Sistema de Educación Pública, comenzó a operar en enero del 2007. Actualmente sirve a más de cuarenta mil

Capítulo 1: Fundamentación teórica

usuarios entre ellos maestros, directores de escuelas, personal de apoyo y administrativo.

Uno de los servicios más útiles del SIE, es el Portal de Padres, mediante el cual los padres o encargados pueden acceder a los expedientes de sus hijos por medio de una página electrónica. En el momento en que el padre o madre se registra con su nombre de usuario y contraseña, el sistema asocia su cuenta de usuario con las cuentas de estudiantes que está autorizado acceder. Entre los servicios que ofrece el nuevo portal de padres se encuentran:

- Información Demográfica del estudiante
- Programa de clases
- Registro de puntuaciones
- Detalles de las evaluaciones académicas y sus calificaciones
- Informes de asistencia [7].

El estudio de este sistema sirve de guía para tener en cuenta a la hora de realizar el registro de información en la universidad, una forma más organizada teniendo en cuenta las esferas evaluada por la UCI, además las personas encargadas con este proceso deben estar logeadas por un problema de seguridad de la información. No se utiliza ya que registra solo información que responde a sus necesidades las cuales no son aplicables en el trabajo.

Sistema Automatizado Informativo para la Comunidad de Residencia Estudiantil en la Universidad de Pinar del Río

El personal directivo de la Comunidad de Residencia Estudiantil en la Universidad de Pinar del Río (UPR) lleva el control de los ingresos y bajas de los estudiantes, la ubicación y el registro de diferentes indicadores como las evaluaciones que se le hacen a los becados en cada semestre y anualmente, las evaluaciones en actividades de cuarterías, guardias, inspecciones a los cuartos, también se controlan sus reconocimientos, sanciones y tratamientos en caso de que el becario padezca de enfermedades [8].

El estudio de este sistema también sirve de guía a la hora de registrar información en las diferentes actividades que se realizan dentro de la universidad, donde se pueden recolectar además las medidas disciplinarias y las distinciones obtenidas por los estudiantes. Este sistema de la UPR no se utiliza ya que este responde a sus necesidades pertinentes las cuales no son aplicables en la UCI.

Sistema de Caracterización Integral del Graduado (SCIG)

Capítulo 1: Fundamentación teórica

Es un sistema desarrollado por el Centro de Innovación de la Calidad de Educación (CICE) en la UCI, con el objetivo de automatizar el proceso de registro para contribuir a la caracterización integral de los estudiantes de quinto año. Este sistema permite el registro de datos generales y resultados obtenidos en las distintas esferas evaluadas en la universidad, mostrando un por ciento de las caracterizaciones realizadas por cada centro, para la toma de decisiones por parte de los directivos que están al frente del proceso. Luego se exporta la plantilla de caracterización en formato PDF para su almacenamiento o impresión. Esta plantilla de caracterización se le entrega a los empleadores para que conozcan las habilidades poseídas por el futuro profesional.

El estudio de este sistema le brinda al trabajo una forma organizada de registrar la información de los estudiantes, pero solo para el último año de la carrera y se necesita uno que pueda ser aplicable a los años restantes. Esta aplicación evalúa las esferas que se tendrán en cuenta en el desarrollo de la investigación, realizarle nuevas funcionalidades que cumplan con la necesidad actual sería muy engorroso debido a que no se integra con el SGU y accede a la información a través de servicios web.

1.4 Proceso de desarrollo ágil con segundo nivel de CMMI (*Capability Maturity Model Integration*)

La ingeniería de *software* desarrolla sus procesos a través de las siguientes fases:



Figura 1: fases del proceso de desarrollo de software

Capítulo 1: Fundamentación teórica

Un proceso de desarrollo de *software* delimita quien está haciendo qué, cómo y cuándo, con el objetivo de lograr la eficiencia en la creación de un *software*. Cuando el proceso es capaz de dar repuestas rápidas y flexibles al cambio sin requerir una gran documentación, se puede decir que es un proceso de desarrollo ágil.

El proceso ágil es un marco de trabajo conceptual de la ingeniería de *software* que promueve iteraciones en el desarrollo a lo largo de todo el ciclo de vida del proyecto. Existen muchos métodos de desarrollo ágil, entre ellos tenemos el Scrum, AUP (*Agile Unified Process*), XP (*Programación Extrema*); la mayoría minimiza riesgos desarrollando *software* en cortos lapsos de tiempo. Entre sus beneficios se encuentran:

- Calendarios y presupuestos predecibles en los proyectos.
- Mejora del ciclo de vida dentro del desarrollo de *software*.
- Mayor productividad.
- Mayor calidad de los productos y servicios que ofrece la universidad a sus clientes y por ende la satisfacción de los mismos.
- Mejorar la moral del personal que labora en el centro.
- Integra disciplinas como sistemas y *software* en un solo marco de trabajo.
- Describe formas efectivas y probadas de hacer las cosas, no es un enfoque radical.

El manifiesto ágil se rige por una serie de principios que hacen flexible la elaboración del *software*, además de tener en cuenta la comunicación con el cliente que es un factor bien importante y la unidad en el equipo de trabajo. A continuación se mencionan algunos de ellos:

- La mayor prioridad es satisfacer al cliente mediante la entrega en tiempo y continúa de *software* con valor.
- Aceptan cambios de requisitos en cualquier etapa del desarrollo, incluso en las tardías.
- Entregan *software* funcional frecuentemente, entre dos semanas y dos meses, con preferencia al periodo de tiempo más corto posible.
- Los responsables del negocio y los desarrolladores trabajan de la mano durante todo el proyecto.
- Los proyectos se desarrollan en torno a individuos motivados, ya que le brindan el entorno y el apoyo que necesitan, confiándoles la ejecución del trabajo.
- El método más eficiente y efectivo de comunicar información al equipo de desarrollo y entre sus

Capítulo 1: Fundamentación teórica

miembros es la conversación cara a cara.

- La medida fundamental del progreso es el *software* funcionando.
- Los procesos ágiles promueven el desarrollo sostenible. Los promotores, desarrolladores y usuarios deben ser capaces de mantener un ritmo constante de forma indefinida.
- Continuamente velan por la excelencia técnica y el buen diseño ya que mejora y aumenta la agilidad [9].

CMMI mejora y evalúa los procesos de desarrollo y mantenimiento de sistemas y productos de *software*, es el modelo de certificación internacional más prestigioso en la ingeniería *software*, se mide a través de cinco niveles el grado de madurez que presentan los procesos. Cada nivel es un escalón bien definido de mejora de proceso y estabiliza una parte importante de los procesos organizacionales, es un estándar de calidad reconocido a nivel mundial. A continuación se mencionan sus niveles de madurez:

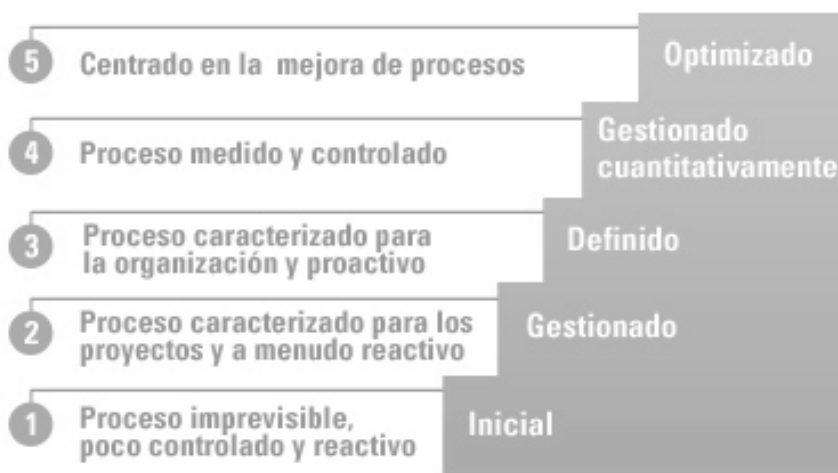


Figura 2: niveles del CMMI

Actualmente la UCI se encuentra inmersa en un proceso de mejora de los procesos que se desarrollan como parte de la construcción del *software* basado en el modelo de CMMI. Este proceso de mejora tiene como objetivo que la universidad alcance el nivel dos del modelo CMMI, lo que la convertiría en la primera institución en el país en alcanzar este nivel [10].

Ciclo de vida del proceso de desarrollo ágil con el segundo nivel del CMMI

Capítulo 1: Fundamentación teórica

- **Estudio preliminar:** es la fase inicial en la cual se realizan las actividades relacionadas con la planeación del proyecto a un alto nivel, permitiendo obtener información sobre el alcance del mismo y realizar estimaciones de tiempo, esfuerzo y costo.
- **Modelado de negocio:** permite conocer los procesos del negocio de una determinada organización, comprende el funcionamiento de este y garantiza que el *software* desarrollado va a cumplir su propósito.
- **Requisitos:** se desarrolla el modelo del sistema y se realiza la especificación de requisitos que incluye los funcionales y no funcionales.
- **Análisis y diseño:** se realiza el análisis y diseño del sistema según los requisitos obtenidos y especificados en la plantilla de especificación de requisitos.
- **Implementación:** según los resultados del análisis previamente realizado se realiza la implementación del sistema.
- **Pruebas internas:** se verifican las versiones que van a ser liberadas y se desarrollan las pruebas de diseño de casos de prueba y lista de chequeo.
- **Pruebas de liberación:** la institución principal encargada de la calidad y certificación de los proyectos valida los mismos antes de ser entregados al usuario.
- **Despliegue:** durante esta fase se procede a la entrega de la solución y capacitación a los trabajadores del sistema, además de realizarse las pruebas de aceptación y las pilotos.
- **Soporte:** durante esta fase y por un tiempo limitado el proyecto ofrecerá un servicio para resolver conflictos y problemas de usabilidad y rendimiento del *software* entregado al cliente, suministrándole actualizaciones y parches a errores.

Lo que se pretende con el nivel 2 de CMMI es conseguir que en los proyectos exista una gestión de los requisitos y que los procesos estén planificados, ejecutados, medidos y controlados. Ya que este garantiza que los mismos sean planeados, documentados, realizados, monitoreados y controlados a nivel de proyectos. El estado de los productos es visible para la administración en puntos de control preestablecidos para que los objetivos se cumplan con éxito.

1.5 Lenguaje de modelado

Lenguaje Unificado de Modelado (UML, por sus siglas en inglés *Unified Modeling Language*). Está consolidado como el lenguaje estándar en el análisis y diseño de sistemas de cómputo. Mediante UML es posible establecerle serie de requisitos y estructuras necesarias para plasmar un sistema de *software* previo al proceso intensivo de escribir código. Es un estándar de la industria del *software*,

Capítulo 1: Fundamentación teórica

pero no solo de la industria, sino, que en general, de cualquier industria que requiera la construcción de modelos como condición previa para el diseño y posteriormente para la construcción de prototipos.

Los objetivos de UML son muchos, pero se pueden sintetizar sus funciones:

- **Visualizar:** permite expresar de una forma gráfica un sistema de forma que otro lo puede entender.
- **Especificar:** permite especificar cuáles son las características de un sistema antes de su construcción.
- **Construir:** a partir de los modelos especificados se pueden construir los sistemas diseñados.
- **Documentar:** los propios elementos gráficos sirven como documentación del sistema desarrollado que pueden servir para su futura revisión [11].

Esta herramienta fue utilizada en la investigación para la realización del modelo de proceso que se desarrolló, dando pie al levantamiento de los requisitos y construcción de las interfaces

1.6 Herramientas y tecnologías

La investigación será desarrollada utilizando las tecnologías y herramientas establecidas por el centro CENIA, el cual establece utilizar como lenguaje de programación PHP5 (*Hypertext Pre-Processor*), GUUD (*Gestión Universitaria Universo Digital Gestión Documental*) como marco de trabajo y PostgreSQL como gestor de base de datos, siguiendo el proceso de desarrollo ágil con segundo nivel de CMMI.

1.6.1 Visual Paradigm 8.0

Es una suite de aplicaciones, herramienta profesional que soporta el ciclo completo de desarrollo de *software*. Soporta el modelado visual con UML que ayuda a una rápida construcción de aplicaciones de calidad, mejores y a un menor costo. Permite generar documentación en HTML (*Hyper Text Markup Language*)/PDF (*formato de documento portátil*) con los diagramas realizados. Permite la generación de bases de datos, conversión de diagramas entidad-relación a tablas de base de datos, mapeo de objetos y relaciones, ingeniería inversa desde gestores de bases de datos. Proporciona una plataforma de modelado colaborativo para el trabajo en equipo. Pueden ver y editar el mismo proyecto, o el esquema de forma simultánea. Todos los cambios se almacenan en el servidor de Visual Paradigm en función de revisión.

Capítulo 1: Fundamentación teórica

El mismo se caracteriza por:

- Presentar una licencia gratuita y comercial
- Estar disponible para múltiples plataformas (Windows, Linux).
- Permitir el uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- Permitir la realización de ingeniería directa e inversa.
- Modelado colaborativo con CVS y Subversión.

1.6.2 Evolus Pencil 1.3

Software de animación y diseño disponible para Mac OS X versión 10.3.9 en adelante, Windows XP o Vista y Linux Ubuntu [12]. Es una extensión de Firefox que se utiliza para el diseño de los prototipos de las interfaces de usuario presentando las siguientes características:

- Brindar un conjunto de componentes de: entradas de texto, botones, tabs y otros componentes que posibilitan la creación de interfaces amigables y de fácil interacción para el usuario.
- Es multi-página, creación simultánea de varios documentos.
- Edición en pantalla de los elementos de texto.
- Permitir exportar imágenes al formato PNG, HTML o PDF.
- Permitir operaciones estándar de dibujo alineado, escalado, rotación, etc.
- Multiplataforma.
- Posibilita a través de las propiedades de los componentes darle estilo al diseño.

1.6.3 Apache JMeter 2.6

Es una aplicación de escritorio (100% Java) de código abierto desarrollada por *Apache Jakarta* que es un proyecto de la *Apache Software Foundation*. Básicamente, fue diseñada para realizar pruebas de carga y medir el rendimiento de sistemas. Esta herramienta se puede utilizar para probar el rendimiento de los recursos estáticos y dinámicos de los sistemas (archivos, Servlets, scripts de Perl, objetos Java, bases de datos y consultas, servidores FTP) y para simular la carga de un servidor, red u objeto con el fin de poner a prueba su resistencia (conocidas como pruebas de *stress*) [13].

Capítulo 1: Fundamentación teórica

1.6.4 Marco de desarrollo

En general, el término *framework* se refiere a una estructura de *software* compuesta de componentes personalizables e intercambiables para el desarrollo de una aplicación. En términos informales, un *framework* se puede considerar como funcionalidades generales a las cuales se les puede agregar las específicas para resolver un problema determinado y obtener una solución informática. GUUD es un marco de trabajo propuesto por el equipo de arquitectura del CENIA. El mismo integra a su vez los *frameworks* CodeIgniter en su versión 1.7.3 y JQuery 1.3.2 en una sola infraestructura, razón por la cual posee las mismas características que estos. En esta integración se incluyen además un conjunto de novedades o mejoras y algunas modificaciones hechas específicamente al CodeIgniter, además de la creación de *plugins* y componentes de interfaz de usuario en JQuery.

CodeIgniter 1.7.3: es un *framework* para el desarrollo de aplicaciones web usando PHP. Este permite el desarrollo de proyectos mucho más rápidos que si se escribiera código desde cero. Provee una rica colección de librerías para las tareas necesarias más comunes. CodeIgniter permite concentrarse en el desarrollo del proyecto en cuestión, minimizando la cantidad de código necesaria para realizar las tareas. Utiliza el patrón de diseño arquitectónico Modelo-Vista-Controlador como paradigma de arquitectura de desarrollo, la cual separa en 3 capas distintas [14].

JQuery 1.3.2: es un nuevo tipo de biblioteca o *framework* de JavaScript que permite simplificar la manera de interactuar con los documentos HTML, permitiendo manejar eventos, desarrollar animaciones y agregar interacción con la tecnología AJAX (*Asynchronous JavaScript and XML*) al sistema. JQuery, al igual que otras librerías, ofrece una serie de funcionalidades basadas en JavaScript que de otra manera requerirían de mucho más código. Es decir, con las funciones propias de esta librería se logran grandes resultados en menos tiempo y espacio. La gran ventaja de JQuery es que permite cambiar el contenido de nuestra página web sin necesidad de recargarla, utilizando DOM (*Document Object Model*) y AJAX de manera extremadamente sencilla gracias a su sintaxis [15].

1.6.5 Sistema gestor de base de datos

Un sistema gestor de base de datos se define como el conjunto de programas que administran y gestionan la información contenida en una base de datos. Ayuda a la definición de los datos, mantenimiento de la integridad de estos dentro de la base de datos, control de la seguridad, privacidad

Capítulo 1: Fundamentación teórica

y manipulación de los datos.

PostgreSQL 8.4.1: distribuye bajo la licencia de PostgreSQL, ofrece muchas ventajas para su compañía o negocio respecto a otros sistemas de bases de datos, entre las que tenemos:

Instalación ilimitada:

Es frecuente que las bases de datos comerciales sean instaladas en más servidores de lo que permite la licencia. Algunos proveedores comerciales consideran a esto la principal fuente de incumplimiento de licencia. Se puede usar PostgreSQL, pues no estaría violando acuerdos de licencia, puesto que no hay costo asociado a la licencia del *software*.

Existen otras ventajas adicionales:

- Modelos de negocios más rentables con instalaciones a gran escala.
- No existe la posibilidad de ser auditado para verificar el cumplimiento de la licencia en ningún momento.
- Flexibilidad para hacer investigación y desarrollo sin necesidad de incurrir en costos adicionales de licenciamiento [16].

PostgreSQL está ampliamente considerado como el sistema de bases de datos de código abierto más avanzado del mundo. Posee muchas características que tradicionalmente solo se podían ver en productos comerciales de alto calibre.

PgAdmin III 1.10.0: licencia BSD, pgAdmin III es una aplicación gráfica para gestionar el gestor de bases de datos PostgreSQL, siendo la más completa y popular con licencia Open Source. Es capaz de gestionar versiones a partir de la PostgreSQL 7.3. Está diseñado para responder a las necesidades de todos los usuarios, desde escribir consultas SQL simples hasta desarrollar bases de datos complejas. El interfaz gráfico soporta todas las características de PostgreSQL y facilita enormemente la administración. La aplicación también incluye un editor SQL con resaltado de sintaxis [17].

1.6.6 Servidor web

Un servidor web es un programa que implementa el protocolo HTTP que pertenece a la capa de aplicación del modelo OSI y está diseñado para transferir lo que llamamos hipertextos, páginas web o páginas HTML textos complejos con enlaces, figuras, formularios, botones y objetos incrustados como

Capítulo 1: Fundamentación teórica

animaciones o reproductores de música y se ejecuta continuamente en un ordenador que responde adecuadamente a las peticiones por parte de un cliente (un navegador web), mediante una página web que se exhibe en el navegador o mostrando el respectivo mensaje si se detectó algún error.

Apache 2.2.2: es el servidor web más utilizado en el mundo. Su configurabilidad, robustez y estabilidad hacen que cada vez millones de servidores reiteren su confianza en este programa. La licencia Apache es una descendiente de la licencia BSD. Esta licencia te permite hacer lo que quieras con el código fuente siempre que se les reconozca su trabajo.

Características de Apache

- Corre en una multitud de sistemas operativos, lo que lo hace prácticamente universal.
- Apache es una tecnología gratuita de código abierto. El hecho de ser gratuita es importante pero no tanto como que se trate de código fuente abierto.
- Es un servidor altamente configurable de diseño modular. Es muy sencillo ampliar las capacidades del servidor web Apache. Actualmente existen muchos módulos para Apache que son adaptables a este, y están ahí para que sean instalados cuando sea necesario.
- Trabaja con gran cantidad de lenguajes entre los cuales se encuentra PHP.
- Permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor.
- Tiene una alta configurabilidad en la creación y gestión de registros. Permite la creación de ficheros de registro a medida del administrador, de este modo se puede tener un mayor control sobre lo que sucede en el servidor [18] .

1.6.7 Entorno integrado de desarrollo «IDE»

NetBeans 6.7.1: el IDE NetBeans es una herramienta para desarrolladores de *software* diseñada para crear con rapidez aplicaciones web, de escritorio y aplicaciones móviles. Es un producto libre y gratuito sin restricciones de uso [19] .

1.6.8 Lenguajes de programación

CSS 2: las hojas de estilo en cascada (Cascading Style Sheets, CSS) son un lenguaje formal usado para definir la presentación de un documento estructurado escrito en HTML o XML (*extensible markup language*). El W3C (*world wide web consortium*), es un consorcio internacional que produce recomendaciones, es el encargado de formular la especificación de las hojas de estilo que servirán de

Capítulo 1: Fundamentación teórica

estándar para los navegadores. La idea que se encuentra detrás del desarrollo de CSS es separar la estructura de un documento de su presentación [20] .

Java Script 1.6: es un lenguaje de programación interpretado, es decir, que no requiere compilación, es utilizado principalmente en páginas web, con una sintaxis semejante a la del lenguaje Java y lenguaje C#. Es un lenguaje orientado a objetos, ya que dispone de Herencia, la cual se realiza siguiendo el paradigma de programación basada en prototipos, ya que las nuevas clases se generan clonando las clases base (prototipos) y extendiendo su funcionalidad. Java Script se ejecuta en el cliente al mismo tiempo que las sentencias van descargándose junto con el código HTML [21] .

PHP 5: es un lenguaje de "código abierto" interpretado, de alto nivel, embebido en páginas HTML y ejecutado en el servidor. Con PHP no se encuentra limitado a resultados en HTML. Entre las habilidades de PHP se incluyen: creación de imágenes, archivos PDF y películas Flash (usando libswf y ming) sobre la marcha. También puede presentar otros resultados, como XHTML y archivos XML. PHP puede auto generar estos archivos y almacenarlos en el sistema de archivos en vez de presentarlos en la pantalla. Quizás la característica más potente y destacable de PHP es su soporte para una gran cantidad de bases de datos. Escribir un interfaz vía web para una base de datos es una tarea simple con PHP [22] .

Entre sus principales ventajas se pueden mencionar:

- Es gratuito e independiente de la plataforma, rápido, con una gran librería de funciones y mucha documentación.
- Permite las técnicas de Programación Orientada a Objeto (POO).
- No requiere definición de tipos de variables, aunque, las mismas se pueden evaluar también por el tipo que estén manejando en tiempo de ejecución.
- Permite la conexión a diferentes tipos de servidores de bases de datos tales como MySQL, PostgreSQL, Oracle, ODBC, DB2, Microsoft SQL Server y SQLite.
- Puede ser desplegado en la mayoría de los servidores Web y en casi todos los sistemas operativos y plataformas sin costo alguno.

1.7 Conclusiones parciales

En el capítulo se ha expuesto el marco informativo y conceptual como resultado de una investigación

Capítulo 1: Fundamentación teórica

llevada a cabo con el objetivo de dar solución al problema propuesto. Se realizó un estudio de diferentes sistemas, lo que permitió conocer las carencias existentes en cuanto a los procesos de caracterización. Los sistemas estudiados se ajustan a las necesidades y características de sus instituciones por lo que no son aplicables a nuestro campo de acción. El estudio de las herramientas y tecnologías establecidas por el centro de CENIA, permitió profundizar los conocimientos necesarios para el desarrollo de un sistema que permita el registro de información para la caracterización de los estudiantes de la UCI, lo cual conlleva a utilizar PHP 5 como lenguaje de programación, GUUD como marco de trabajo, el SGBD PostgreSQL 8.4.1 y el servidor web Apache 2.0.

Capítulo 2: Análisis y diseño del sistema

2.1 Introducción

En el presente capítulo se hace una propuesta general del sistema a desarrollar, después de haber realizado un análisis crítico de los procesos de negocio que tienen lugar en la universidad. Se obtiene el modelo de negocio, los requerimientos funcionales y no funcionales, se realizará una pequeña descripción de cómo se llevan actualmente los procesos involucrados en el campo de acción, describiendo además la arquitectura, patrones a utilizar, diagrama de despliegue y el modelo de datos correspondiente con el proceso de desarrollo ágil.

2.2 Modelo del negocio

En el trabajo no se realiza el modelo de procesos debido a la ausencia de un negocio bien definido, por lo que se hizo necesario realizar una conceptualización o modelo de dominio con los principales conceptos definidos.

Los modelos de dominio pueden utilizarse para capturar y expresar el entendimiento ganado en un área de bajo análisis como paso previo al diseño de un sistema, ya sea de *software* o de otro tipo. Similares a los mapas mentales utilizados en el aprendizaje, el modelo de dominio es utilizado por el analista como un medio para comprender el sector industrial o de negocios al cual el sistema va a servir [23].

El proceso para su elaboración tiene tres pasos:

1. Identificar las clases conceptuales.
2. Dibujarlas en un diagrama de clases.
3. Añadir relaciones y atributos.

Luego seguir los pasos para la elaboración del modelo de dominio se detectaron 11 clases conceptuales las cuales se describen a continuación:

- **Sistema de Gestión Universitaria (SGU):** se encarga de informatizar e integrar los procesos sustantivos de la universidad, entre sus áreas de procesos se encuentran pregrado, postgrado,

Capítulo 2: Análisis y diseño del sistema

residencia, cooperación, ingreso, investigación, producción, laboratorios, biblioteca, extensión universitaria, teleformación y egreso.

- **Centro Innovación y Calidad de la Educación (CICE):** es el encargado de contribuir a elevar la calidad del proceso de formación integral de los estudiantes en la UCI.
- **Registro estudiantil:** permite registrar información del estudiante en las diferentes esferas evaluadas por la universidad, además de tener en cuenta su trayectoria FEU-UJC, indisciplinas cometidas y las competencias técnicas adquiridas.
- **Información de formación:** permite registrar todas las actividades docentes que realice el estudiante durante la carrera.
- **Información de extensión:** permite registrar las actividades extensionistas del estudiante durante la carrera.
- **Información de investigación:** permite registrar las actividades investigativas que realice el estudiante durante la carrera.
- **Información de producción:** permite registrar las actividades productivas que realice el estudiante durante la carrera.
- **Información de trayectoria FEU-UJC:** permite registrar las actividades de la FEU durante toda la carrera, además de brindar la posibilidad de registrar las de la UJC en caso que el estudiante pertenezca a esta organización.
- **Información de indisciplinas:** permite registrar las faltas disciplinarias cometidas por el estudiante durante la carrera.
- **Información de competencias técnicas:** permite registrar las competencias técnicas que obtenga el estudiante junto al nivel de dominio de las mismas durante toda la carrera.
- **Usuario:** es quien gestiona todas las acciones del registro de información de los estudiantes de la UCI.

Diagrama de clases del dominio

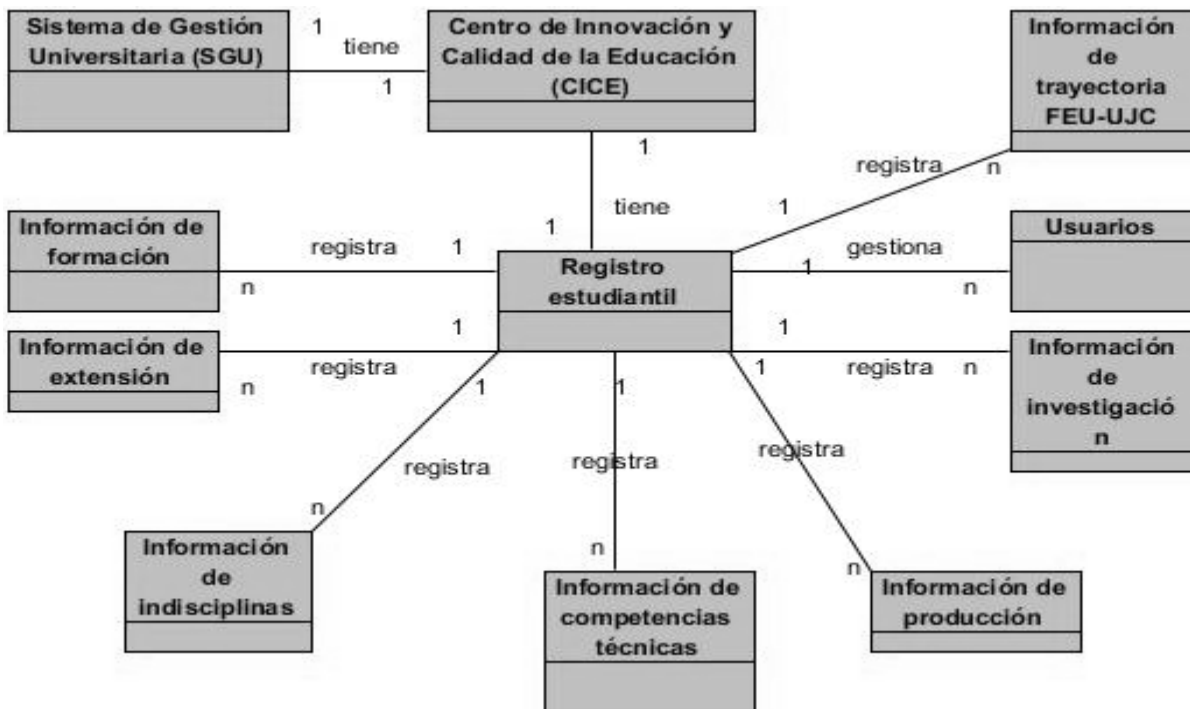


Figura 3: modelo de dominio

2.3 Propuesta de solución

Luego de detectar la ausencia de un sistema capaz de registrar la información relacionada a las distintas esferas evaluadas que comprende el nivel superior de la UCI para todos los años, se propone desarrollar el módulo Registro de información para la caracterización de los estudiantes en la Universidad de las Ciencias Informáticas, brindando las siguientes soluciones:

- Asignación de responsabilidades a las personas encargadas de realizar el registro de información a los estudiantes.
- Realización del registro de caracterización en las diferentes esferas que se evalúan en la universidad: formación, extensión universitaria, investigación y producción, además de tener en cuenta la trayectoria FEU-UJC, faltas y medidas disciplinarias que pueda obtener el estudiante y las competencias técnicas que adquiera y con qué nivel. (El Ministerio de Educación Superior (MES) define como modelo formativo la “investigación”, “extensión” y “formación”, pero la UCI tiene sus particularidades ya que este proceso es centrado en el aprendizaje de los estudiantes

Capítulo 2: Análisis y diseño del sistema

con el objetivo de formar profesionales altamente calificados. La universidad tiene en su modelo de formación la vinculación estudio trabajo y la integración de los procesos de producción e investigación).

- Gestión del registro de información durante toda la carrera con el objetivo de tener un seguimiento real de las actividades realizadas por los estudiantes.
- Acceso a información registrada en años anteriores que permita el control y seguimiento de los estudiantes.

La solución debe desarrollarse cumpliendo cada una de las expectativas del cliente para lograr su satisfacción, además la aplicación debe ser entendible y de fácil manejo para las personas involucradas en el proceso de registro.

2.4 Definición de los requisitos funcionales y no funcionales

El proceso de recopilar, analizar y verificar las necesidades del cliente o usuario para un sistema es llamado Ingeniería de Requerimientos (IR), este tiene como meta entregar una especificación de requisitos de *software* correcta y completa.

La IR ayuda a los ingenieros de *software* a entender mejor el problema en cuya solución trabajarán. Incluye el conjunto de tareas que conducen a comprender cuál será el impacto del *software* sobre el negocio, qué es lo que el cliente quiere y cómo interactuarán los usuarios finales con el *software* [24].

2.4.1 Requisitos funcionales

Los requisitos funcionales de un sistema dicen que es lo que debe hacer el sistema. Los mismos obedecen al tipo de *software* que se desarrolle, a los posibles usuarios del *software* y al enfoque general de la organización tomado al redactar los requerimientos. Puntualizan las funciones que el sistema pueda realizar, los cambios que el sistema ocasiona sobre las entradas para generar las salidas

2.4.2 Definición de los requisitos funcionales

Los requisitos funcionales son capacidades o condiciones que el sistema debe realizar, es decir, define qué es lo que el sistema debe hacer. En la investigación se clasificaron los requerimientos teniendo en cuenta su complejidad en alta, media, baja.

Capítulo 2: Análisis y diseño del sistema

Requisitos	Complejidad
1. Buscar estudiante	Alta
2. Ver detalles de estudiante	Alta
3. Registrar información de formación	Alta
4. Registrar información de extensión	Alta
5. Registrar información de investigación	Alta
6. Registrar información de producción	Alta
7. Registrar información de trayectoria FEU-UJC	Alta
8. Registrar información de faltas disciplinarias	Alta
9. Registrar información de competencias técnicas	Alta
10. Mostrar evento	Media
11. Crear evento	Alta
12. Modificar evento	Alta
13. Ver detalles de evento	Baja
14. Mostrar tipos de evento	Media
15. Crear tipos de evento	Alta
16. Modificar tipos de evento	Alta
17. Ver detalles de tipos de evento	Baja
18. Mostrar actividades	Media
19. Crear actividades	Alta
20. Modificar actividades	Alta
21. Ver detalles de actividades	Baja
22. Mostrar tipos de actividad	Media
23. Crear tipos de actividad	Alta
24. Modificar tipos de actividad	Alta

Capítulo 2: Análisis y diseño del sistema

25. Ver detalles de tipos de actividad	Baja
26. Mostrar niveles	Media
27. Crear nivel	Alta
28. Modificar nivel	Alta
29. Ver detalles de nivel	Baja
30. Mostrar responsabilidades	Media
31. Crear responsabilidad	Alta
32. Modificar responsabilidad	Alta
33. Ver detalles de responsabilidad	Baja
34. Mostrar roles	Media
35. Crear rol	Alta
36. Modificar rol	Alta
37. Ver detalles de rol	Baja
38. Mostrar nivel de rol	Media
39. Crear nivel de rol	Alta
40. Modificar nivel de rol	Alta
41. Ver detalles de nivel de rol	Baja
42. Mostrar resultados	Media
43. Crear resultado	Alta
44. Modificar resultado	Alta
45. Ver detalles de resultado	Baja
46. Mostrar evaluaciones	Media
47. Crear evaluación	Alta
48. Modificar evaluación	Alta
49. Ver detalles de evaluación	Baja

Capítulo 2: Análisis y diseño del sistema

50. Mostrar actitud ante el estudio	Media
51. Crear actitud ante el estudio	Alta
52. Modificar actitud ante el estudio	Alta
53. Ver detalles de actitud ante el estudio	Baja
54. Mostrar trayectorias	Media
55. Crear trayectoria	Alta
56. Modificar trayectoria	Alta
57. Ver detalles de trayectoria	Baja
58. Mostrar tipo de distinción	Media
59. Crear tipo de distinción	Alta
60. Modificar tipo de distinción	Alta
61. Ver detalles de tipo de distinción	Baja
62. Mostrar lenguajes	Media
63. Crear lenguaje	Alta
64. Modificar lenguaje	Alta
65. Ver detalles de lenguaje	Baja
66. Mostrar herramientas de desarrollo de <i>software</i>	Media
67. Crear herramienta	Alta
68. Modificar herramienta	Alta
69. Ver detalles de herramienta	Baja
70. Mostrar metodologías	Media
71. Crear metodologías	Alta
72. Modificar metodologías	Alta
73. Ver detalles de metodologías	Baja
74. Mostrar gestores de bases de datos	Media

Capítulo 2: Análisis y diseño del sistema

75. Crear gestor de base de datos	Alta
76. Modificar gestor de base de datos	Alta
77. Ver detalles de gestor de base de datos	Baja
78. Mostrar plataformas de desarrollo	Media
79. Crear plataforma de desarrollo	Alta
80. Modificar plataforma de desarrollo	Alta
81. Ver detalles de plataforma de desarrollo	Baja
82. Asociar persona a proyecto	Alta
83. Eliminar persona a proyecto	Alta
84. Mostrar proyectos	Media
85. Ver detalles de proyecto	Baja
86. Mostrar personas	Media

Tabla 1: requisitos funcionales

2.4.3 Especificación de los requisitos funcionales

La especificación de requisitos permite controlar el alcance de los objetivos definidos en el proyecto, ya que reflejan las necesidades del cliente, y con ella misma se verifica si se están cumpliendo los objetivos inicialmente trazados.

Nº	Nombre	Descripción	Complejidad	Prioridad
RFL1	Buscar estudiante	Permite buscar el o los estudiante(s) que se les realizará el registro de información en las distintas áreas de la universidad a través de los diferentes criterios de búsquedas que aparecen (nombre, apellido, usuario, CI, solapín) y los filtros (grupo, año	Alta	Muy Alta

Capítulo 2: Análisis y diseño del sistema

		<p>académico, facultad, provincia, estado docente y sexo). Se muestra el listado de estudiantes con la foto, el nombre, los apellidos y las opciones en el área de íconos internos de ver detalles de, Información de Formación, Extensión, Investigación, Producción, Trayectoria FEU, Faltas disciplinarias y Competencias técnicas.</p> <p>Se muestra un encabezado de páginas que visualiza la cantidad de estudiantes a mostrar en una página.</p> <p>En el borde inferior aparece la página en la que te encuentras y la cantidad de resultados encontrados.</p>		
Prototipo				

Capítulo 2: Análisis y diseño del sistema

Buscar estudiante

Filtro de búsqueda

Seleccione

Cantidad por página

Nombre(s) y Apellidos

<<
<
Página 1 de 1
>
>>
Resultados encontrados 5

Campos	Tipos de Datos	Reglas o Restricciones
Buscador	Alfanuméricos	Solo se puede buscar por los criterios (nombre, usuario, solapin y CI) que se brindan en la aplicación.
Filtro de búsqueda	VarChar	Solo se puede buscar por los filtros (grupo, facultad, provincia, estado docente y sexo) de búsqueda que se brindan en la aplicación.
Observaciones	<ul style="list-style-type: none"> Interactúa con esta acción el profesor del Grupo académico que le corresponde. El listado de estudiante debe aparecer ordenado alfabéticamente primer apellido, segundo apellido, primer nombre, segundo nombre. Cuando se encuentren resultados vacíos se muestran los resultados encontrados en cero. 	

Tabla 2: buscar estudiante

Capítulo 2: Análisis y diseño del sistema

Nº	Nombre	Descripción	Complejidad	Prioridad
RF3	Registrar información de formación.	<p>El requisito permite registrar información de formación.</p> <ul style="list-style-type: none">• El usuario introduce los datos: Actitud ante el estudio, Evaluación en la REM, Evaluación en el TSU, Nombre de evento, Nivel, Resultado.• El sistema permite asociar más de un evento mostrando un listado de los mismos con la información: Nombre del evento, Nivel, Resultados.	Alta	Muy Alta

Prototipo

Registrar información de Formación



Actitud ante el estudio *	Evaluación en la REM *	Evaluación en el TSU *
<input type="text" value="Seleccione"/>	<input type="text" value="Seleccione"/>	<input type="text" value="Seleccione"/>
Nombre de evento	Nivel	Resultado
<input type="text" value="Seleccione"/>	<input type="text" value="Seleccione"/>	<input type="text" value="Seleccione"/>

Nombre evento	Nivel	Resultados
Copa Miguel de Matemática	UCI	Destacado
Copa Pascal	Facultad	Relevante

Capítulo 2: Análisis y diseño del sistema

Campos	Tipos de Datos	Reglas o Restricciones
Actitud ante el estudio	Varchar	El campo debe ser llenado obligatoriamente El campo es de selección
Evaluación en la REM	Varchar	El campo debe ser llenado obligatoriamente El campo es de selección
Evaluación en el TSU	Varchar	El campo debe ser llenado obligatoriamente El campo es de selección
Nombre del evento	Varchar	El campo es de selección
Nivel	Varchar	El campo es de selección
Resultado	Varchar	El campo es de selección
Observaciones		<ul style="list-style-type: none">• El Usuario debe estar autenticado en el sistema.• En caso de que no ocurran errores el sistema mostrará un mensaje de confirmación “El elemento ha sido creado satisfactoriamente”• En caso que el evento exista se muestra un mensaje de error: “El elemento ya existe”.• En caso de cancelar la acción se muestra un mensaje de advertencia “¿Está seguro de realizar la acción?”.• En caso que se deje un campo de los obligatorios vacío se muestra en forma de globo un mensaje en rojo “Campo requerido” encima del campo que debe ser llenado obligatoriamente.• Si se asocia satisfactoriamente el evento de formación el sistema mostrará un mensaje de información “La asociación se ha realizado satisfactoriamente” y le permitirá al usuario asociar un nuevo evento de formación

Tabla 3: crear registro de información de formación

2.5 Requisitos no funcionales

Los requisitos no funcionales, como su nombre sugiere, son aquellos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a propiedades emergentes de este como la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento [25].

Capítulo 2: Análisis y diseño del sistema

Requisitos no funcionales	
Usabilidad	
RNF 1	El sistema debe presentar una interfaz amigable que permita la fácil interacción con el mismo por parte de los usuarios, los cuales deben poder acceder de manera rápida y efectiva a la información solicitada. El sistema debe permitir un fácil aprendizaje para los usuarios, de modo que posibilite en estos una rápida adaptación.
RNF 2	El sistema debe adaptarse al lenguaje y términos utilizados por los clientes en la rama abordada con vista a una mayor comprensión por su parte sobre la herramienta de trabajo.
RNF 3	Diferenciar las interfaces y opciones para los usuarios que accedan al sistema según los diferentes roles que estos tengan dentro del mismo.
RNF 4	El sistema debe presentar una serie de menús tanto laterales como en barras horizontales de iconos que permitan el acceso rápido a la información por parte de los usuarios, aprovechando así las potencialidades de estas estructuras.
Seguridad	
RNF 5	Políticas de seguridad por usuarios y roles: El sistema debe contar con un grupo de políticas de accesibilidad a las diferentes funcionalidades del mismo en dependencia del nivel de autorización que presente un usuario determinado.
RNF 6	Registro sistemáticos de incidencias: El sistema debe ser capaz de registrar el accionar del usuario, así como permitir auditorías y exámenes de las trazas tanto en tiempo real como en históricos.
Disponibilidad	
RNF 7	El sistema estará disponible las 24 horas del día y los siete días de la semana.
Eficiencia	
RNF 8	El sistema debe soportar un tiempo de respuesta menor o igual a 5 segundos, para 1000 usuarios conectados y en una máquina con no menos de 1 GB de RAM.
RNF 9	El sistema debe soportar una conexión simultánea de más de 100 usuarios.
Soporte	
RNF 10	El sistema contará con un grupo de soporte y asesoría al cliente.

Capítulo 2: Análisis y diseño del sistema

RNF 11	El sistema brinda como apoyo una ayuda contextual en la cual se refleja detalladamente la explicación de cada una de las pantallas con sus respectivas funcionalidades.
Restricciones de diseño	
RNF 12	El lenguaje de programación deberá ser PHP5 o superior
RNF 13	El marco de trabajo base de desarrollo que se utilizará es: GUDD
RNF 14	Como Entorno de Desarrollo Integrado se empleará Netbeans 7.1
RNF 15	Como servidor Web se explotará Apache 2.0
RNF 16	El sistema gestor de bases de datos deberá ser PostgreSQL 8.4.1.
RNF 17	El diseño de la base de datos se realizará con Visual Paradigm 8.0
RNF 18	El sistema operativo a utilizar en el entorno de desarrollo deberá ser: GNU Linux.
RNF 19	El repositorio principal, el entorno de prueba y el servidor de base de datos estarán montados sobre Ubuntu Server 10.04 o superior.
Requisitos para la documentación de usuarios en línea y ayuda del sistema	
RNF 20	El sistema deberá presentar un manual de usuario, permitiendo con ello un correcto uso de sus funcionalidades y brindarle al usuario una mayor experiencia del trabajo con el mismo.
RNF 22	Se precisa que la documentación del sistema esté actualizada en todos los aspectos, fases de trabajo y ciclos de desarrollo del mismo, permitiendo con ello un respaldo tanto ingenieril como legal del desarrollo de dicho sistema.
Componentes comprados	
RNF 23	Para el desarrollo del sistema no fue necesario comprar ningún componente.
Interfaces	
RNF 24	La interfaz interna estará determinada por los desarrolladores, construyendo así una vista escalable de las clases o agrupaciones de clases que permitirán un mejor encapsulamiento de las funcionalidades y una mayor abstracción modular del sistema.
Interfaces hardware	

Capítulo 2: Análisis y diseño del sistema

RNF 25	La comunicación entre el cliente y el servidor de aplicaciones se lleva a través del protocolo HTTPS.
Requisitos legales, de derecho de autor y otros	
RNF 26	El sistema debe ser sometido a un análisis legal por parte de los abogados y personal autorizado con vistas a declarar su autenticidad y evitar restricciones legales para su uso y comercialización; así mismo se debe proceder a una evaluación y certificación por parte del cliente del producto.
Requisitos de hardware	
RNF 27	Para el desarrollo se requiere de una PC Intel Pentium 4 o superior, CPU 3GHZ o superior, 512 MB RAM o superior, 160 GB HDD o superior
RNF 28	Para la explotación del cliente se requiere de una PC Pentium 3 o superior, CPU 133 MHZ o superior, 256 RAM mínimo 512 RAM recomendada o superior.
RNF 29	Para la explotación del servidor se requiere de un CPU Dual Core 2.0 GHZ o superior, memoria RAM de 4 GB (recomendado 6 GB), 250 GB HDD.
Estándares aplicables	
RNF 30	Referirse al documento de arquitectura: CENIA_PRE_ADASP-v1.0 (en el mismo se especifica los requisitos de estándares aplicables).

Tabla 4: requisitos no funcionales

2.6 Técnicas de obtención de requisitos.

Las técnicas de obtención de requisitos son aquellas que permiten comprender el dominio del sistema, buscar y recolectar información para definir sus límites y restricciones, e identificar a las personas interesadas en el sistema. Existen diferentes tipos de pruebas, entre ellas tenemos las de tormentas de ideas cuestionarios, revisión de la documentación técnica, ingeniería inversa entre otras. En el trabajo se aplicaron las entrevistas y los talleres estructurados permitiendo obtener una colección y clasificación de los requerimientos del sistema, mediante la participación de los clientes y usuarios.

Entrevistas: se realizaron reuniones entre analistas y cliente que permitió entender el dominio del problema y sus necesidades. Esta se basó en un formato de preguntas y respuestas, buscando obtener las opiniones de los clientes o usuarios entrevistados, sus opiniones del sistema, sus metas

Capítulo 2: Análisis y diseño del sistema

personales, de la organización y de los procedimientos informales. Estas se realizaron tanto personal como en grupo.

Prototipos: un prototipo es una versión inicial de un sistema de *software* que se utiliza para demostrar los conceptos, probar las opciones de diseño y de forma general enterarse más acerca del problema y sus posibles soluciones. Estos fueron mostrados al cliente, quienes proporcionaron los requerimientos adicionales. Se puede cambiar entonces la aplicación, volver a mostrarla al cliente y así sucesivamente. Este proceso repetitivo continúa durante un determinado número de iteraciones o hasta que el producto cumple con las necesidades de negocio más importantes.

2.7 Modelo físico de la Base de Datos

Un modelo de datos es un lenguaje utilizado para la descripción de una base de datos. Por lo general, un modelo de datos permite describir las estructuras de datos de la base (el tipo de los datos que incluye la base y la forma en que se relacionan), las restricciones de integridad (las condiciones que los datos deben cumplir para reflejar correctamente la realidad deseada) y las operaciones de manipulación de los datos (agregado, borrado, modificación y recuperación de los datos de la base)

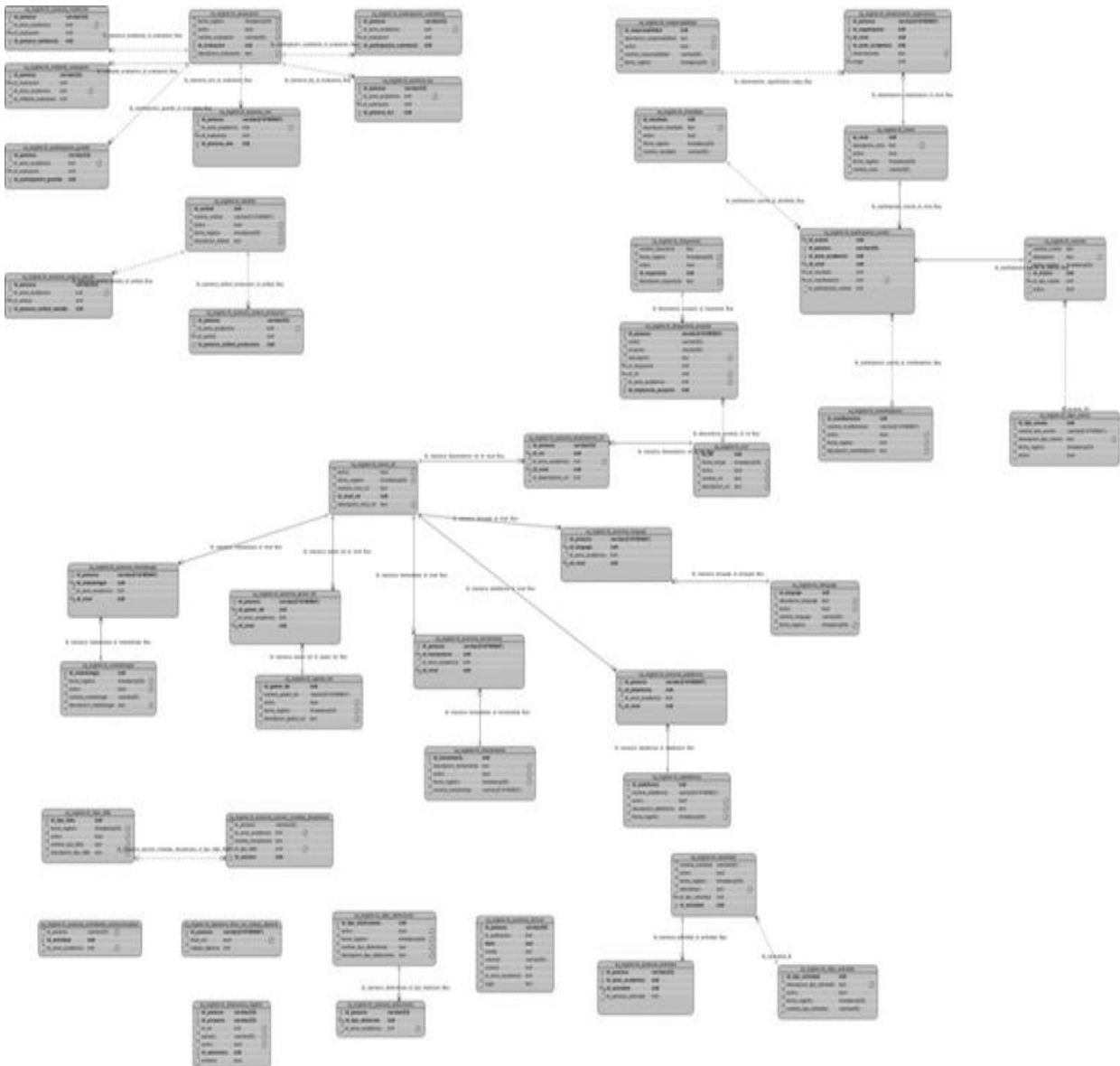


Figura 4: modelo físico de la base de datos

Para mejor visión del diagrama ver Anexo#1

2.8 Estilo arquitectónico

El modelo cliente-servidor es una arquitectura donde el sistema se organiza como un conjunto de servicios y servidores asociados, más unos clientes que acceden y usan dichos servicios [25]. En la arquitectura Cliente/Servidor coinciden una serie de aplicaciones basadas en dos categorías que cumplen funciones desiguales, una solicita servicios y la otra los ofrece. Estas a su vez, consiguen

Capítulo 2: Análisis y diseño del sistema

realizar actividades en forma conjunta e independientemente y son llamadas categorías cliente y servidor.

Características del cliente:

- Es quien comienza las solicitudes o peticiones, por lo que tienen un papel activo en la comunicación.
- Espera y recibe las respuestas del servidor.
- Por lo general, puede conectarse a varios servidores a la vez.

Características del servidor:

- Al iniciarse esperan a que lleguen las solicitudes de los clientes, por lo que realizan un papel pasivo.
- Después de la recepción de una solicitud, la procesan y le envían la respuesta al cliente.
- Generalmente permiten conexiones desde un gran número de clientes (en ocasiones el número de peticiones puede estar limitado).
- Normalmente no interactúan directamente con los usuarios finales.

2.9 Patrón de arquitectura

La arquitectura de *Software*, denominada además como arquitectura lógica, consiste en un conjunto de patrones y abstracciones coherentes que proporcionan el marco de referencia necesario para guiar la construcción del *software* para un sistema de información. Establece los fundamentos para que analistas, diseñadores, programadores, entre otros, trabajen en una línea común que permita alcanzar los objetivos del sistema de información, cubriendo todas las necesidades. Además, define de manera abstracta los componentes que llevan a cabo alguna tarea de computación, sus interfaces y la comunicación entre ellos. Toda arquitectura debe poderse implementar en una arquitectura física, que consiste simplemente en determinar qué computadora tendrá asignada cada tarea.

La arquitectura de *software*, tiene que ver con el diseño y la implementación de estructuras de *software* de alto nivel. Es el resultado de ensamblar un cierto número de elementos arquitectónicos de forma adecuada para satisfacer la mayor funcionalidad y requerimientos de desempeño de un sistema, así como requerimientos no funcionales, como la confiabilidad, escalabilidad, portabilidad, y disponibilidad.

Capítulo 2: Análisis y diseño del sistema

MVC (Modelo Vista Controlador): es un patrón de diseño de arquitectura de *software* que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. Este patrón se ve frecuentemente en aplicaciones Web, donde la vista es la página HTML y el código que provee de datos dinámicos a la página

[26] .

Modelo: esta es la representación específica del dominio de la información sobre la cual funciona la aplicación. El modelo es otra forma de llamar a la capa de dominio.

- **Vista:** se encarga de presentar la interfaz al usuario, esto es típicamente HTML, aunque pueden existir otro tipo de vistas.
- **Controlador:** este responde a eventos, usualmente acciones del usuario e invoca cambios en el modelo y probablemente en la vista.

El primer paso en el ciclo de vida del patrón **MVC** empieza cuando el usuario hace una solicitud al controlador con información sobre lo que el usuario desea realizar. El controlador decide quién debe delegar la tarea y es ahí donde el modelo empieza su trabajo. En esta etapa, el Modelo se encarga de realizar operaciones sobre la información que maneja para cumplir lo que le solicita el controlador. Una vez que termina su labor, le regresa al Controlador la información resultante de sus operaciones el cual redirige a la Vista. La Vista se encarga de transformar los datos en información visualmente entendible para el usuario. Finalmente, la representación gráfica es transmitida de regreso al Controlador y este se encarga de transmitírsela al usuario. El ciclo entero puede comenzar de nuevo si el usuario así lo requiere.

2.10 Patrones de diseño

Los patrones de diseño son soluciones simples y elegantes a problemas específicos y comunes del diseño orientado a objetos. Son soluciones basadas en la experiencia y que se ha demostrado que funcionan. Ayudan a evitar que los cambios en el sistema se realicen de una forma concreta, de manera que se afecte lo menos posible [27].

Contribuyen a reutilizar diseño gráfico, identificando aspectos claves de la estructura de un diseño que puede ser aplicado en una gran cantidad de situaciones. La importancia de la reutilización del diseño no es despreciable, ya que esta nos provee de numerosas ventajas: reduce los esfuerzos de desarrollo

Capítulo 2: Análisis y diseño del sistema

y mantenimiento, mejora la seguridad informática, eficiencia y consistencia de nuestros diseños, y nos proporciona un considerable ahorro en la inversión. Mejoran la flexibilidad, modularidad y extensibilidad, factores internos e íntimamente relacionados con la calidad percibida por el usuario.

Patrones GoF (*Gang of Four*)

Dentro de los patrones Gof se encuentran los de creación y comportamiento, a continuación se mostrarán en que se basan cada uno de ellos:

Creación

- **Fábrica abstracta** (Abstract Factory): permite trabajar con objetos de distintas familias, de manera que las familias no se mezclen entre sí y haciendo transparente el tipo de familia concreta que se esté usando.
- **Instancia única** (Singleton): garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia.

Comportamiento

- **Mediador**: define un objeto que coordine la comunicación entre objetos de distintas clases, pero que funcionan como un conjunto.
- **Observador**: define una dependencia de uno-a-muchos entre objetos, de forma que cuando un objeto cambie de estado se notifique y actualicen automáticamente los que dependan de él.

Patrones GRASP (*Greedy Randomized Adaptive Search Procedures*)

Los patrones de asignación de responsabilidades «**GRASP**», dan la medida de un refinamiento del diseño.

- **Experto**: asignar una responsabilidad al experto en información, la clase que cuenta con la información necesaria para cumplir la responsabilidad.
- **Creador**: asignarle a la clase B la responsabilidad de crear una instancia de clase A.
- **Controlador**: asignar la responsabilidad del manejo de un mensaje de los eventos de un sistema a una clase.
- **Bajo acoplamiento**: asignar una responsabilidad para mantener bajo acoplamiento. El grado de acoplamiento no puede considerarse aisladamente de otros principios como Experto y Alta

Capítulo 2: Análisis y diseño del sistema

Cohesión. Sin embargo, es un factor a considerar cuando se intente mejorar el diseño.

- **Alta Cohesión:** asignar una responsabilidad, de modo que la cohesión siga siendo alta.

2.11 Aplicación de los patrones

GOF

Abstract Factory: en el módulo seguridad, en la librería `fabrica_ma_lib`, que se encarga de crear los objetos de los modos de autenticación (`ma`) que heredan de la clase `autenticacion_lib`, que son `ma servicio web`, `ma base de datos`, `ma ldap` y `ma open ldap`.

Singleton:

- Todas las clases controladoras, son instancias únicas
- La `loc` para la interacción entre módulos.

Mediador: las librerías que funcionan como mediadoras entre las clases controladoras y las modelo o acceso a datos.

Observador: en la clase `loader` que es el objeto `load` de las clases controladoras, encargada de cargar los elementos del marco de trabajo, dígame con esto librerías, modelos y se encarga de actualizar la controladora instanciada.

GRASP

Experto: se evidencia en las clases librerías, que son las que cuentan con la información necesaria para cumplir las responsabilidades sobre los elementos del negocio.

Creador: en la clase `loader` que es el objeto `load` de las clases controladoras, encargada de cargar los elementos del marco de trabajo, dígame librerías, modelos. En el módulo seguridad, en la librería `fabrica_ma_lib`, que se encarga de crear los objetos de los modos de autenticación (`ma`).

Controlador: las clases controladoras que son las encargadas de obtener datos y enviarlos a las librerías y las vistas.

Capítulo 2: Análisis y diseño del sistema

Bajo acoplamiento y Alta Cohesión: la propia implementación de CodeIgniter contiene estos patrones nivelados, ya que permite el uso de los componentes de forma individual, evidenciando el bajo acoplamiento, así como la dependencia entre ellos o alta cohesión.

2.12 Modelo de despliegue.

El Modelo de Despliegue muestra la disposición física de los distintos nodos que componen el sistema. Representa la disposición de las instancias de componentes de ejecución en instancias de nodos conectados por enlaces de comunicación. Sobre el modelo de despliegue debe hacerse las siguientes observaciones:

- Los nodos poseen relaciones que son medios de comunicación entre ellos, tales como TCP/IP, HTTPS, USB.
- Cada nodo representa un recurso de cómputo, puede ser un procesador o un dispositivo hardware.

Seguidamente se muestra el modelo de despliegue del sistema a desarrollar:

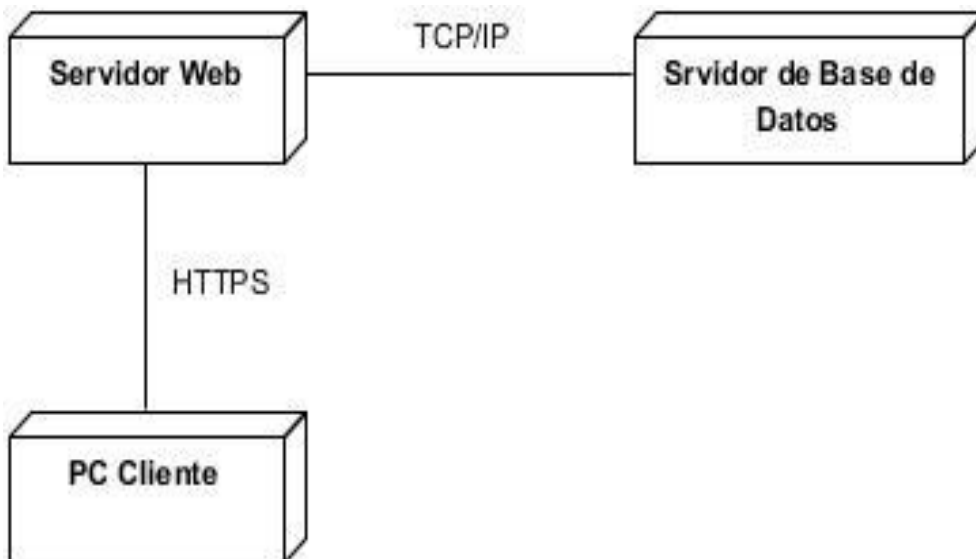


Figura 5: diagrama de despliegue

2.13 Distribución de las funcionalidades del módulo

A continuación se mostrará el mapa de navegación en el cual se podrán observar las funcionalidades del módulo Registro del subsistema CICE, donde se agrupan según las opciones que se gestionan.

Capítulo 2: Análisis y diseño del sistema

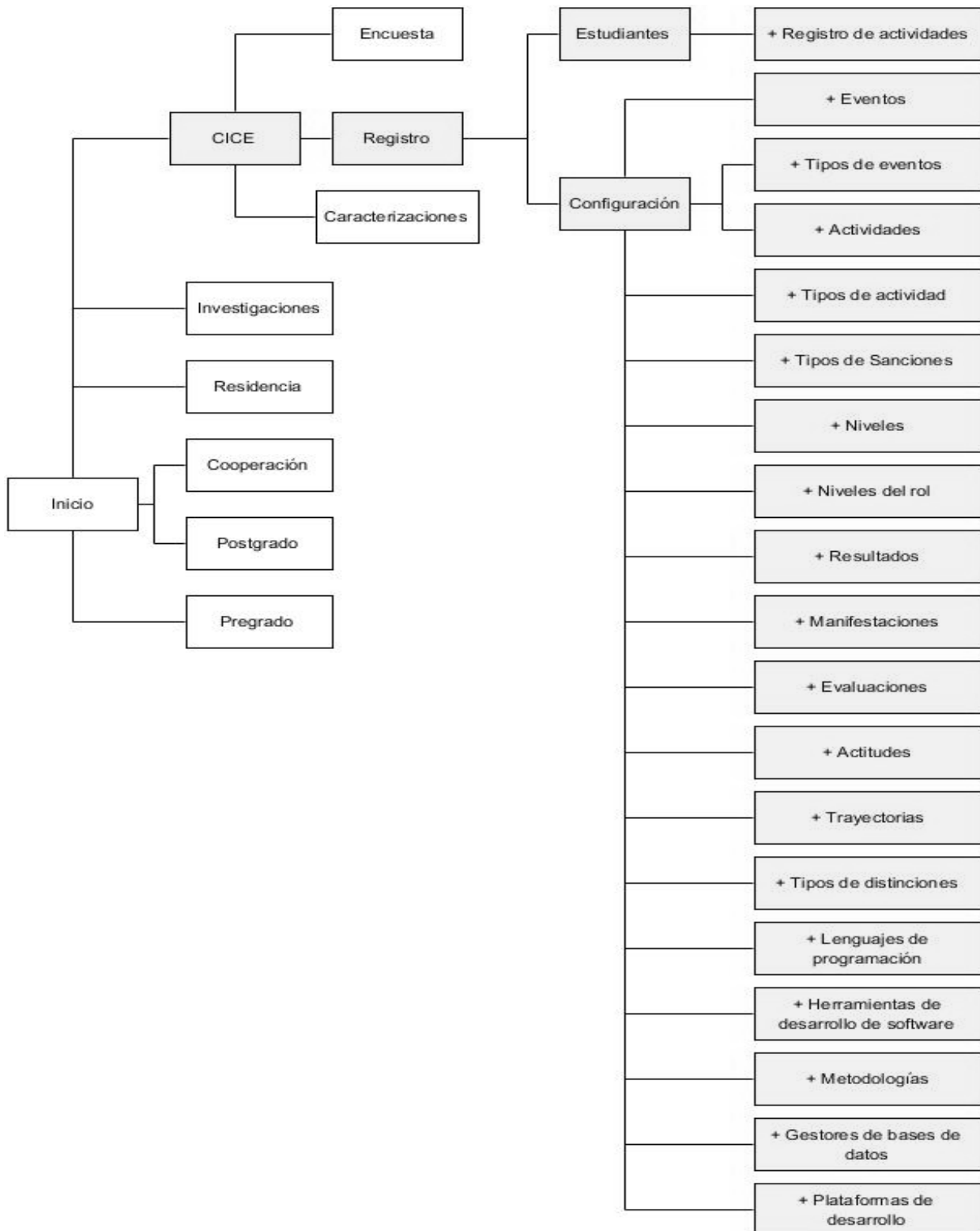


Figura 6: mapa de navegación

Capítulo 2: Análisis y diseño del sistema

2.14 Integración de la propuesta de solución al Sistema de Gestión Académica de Pregrado.

El CICE, se encuentra integrado al SGU, lo que permite el uso de las facilidades provistas por dicha integración, como la reutilización de varios componentes, la centralización de archivos e información y la eliminación de redundancias en el código. La solución propuesta para el módulo de Registro del sistema CICE, precisamente utiliza elementos necesarios de otros módulos del sistema y del núcleo de SGU, de ahí que no sea necesario implementar un gran número de funcionalidades, además tributará con información para el buen funcionamiento de otros módulos.

2.14.1 Seguridad

Garantiza el acceso a la información dados los niveles de privilegio de cada usuario, haciendo uso de la arquitectura sobre la cual está desarrollado el sistema. En Registro la seguridad permitirá que los profesores guías, tutores u otros involucrados en el proceso sean capaces de acceder a las funcionalidades en las que tengan establecidos los permisos, evitando el mal uso, corrupción y modificación de la información por personas no autorizadas.

2.14.2 Configuración

La configuración del sistema, también gestionada de forma centralizada, facilita un mejor control sobre los posibles cambios que pueden ocurrir en el sistema, lo cual provee robustez y adaptabilidad al mismo. Dicho módulo permite gestionar la información de cada sistema, sus módulos y las funcionalidades asociadas.

2.14.1 Trazas

Gestiona todo lo referente a las incidencias de un usuario sobre el sistema, registrando el usuario, la acción realizada y el momento en que se ejecutó. Las acciones de los usuarios sobre el módulo que se propone serán almacenadas por este módulo.

2.14.2 Personal y Secretaría

Se obtiene toda información necesaria de los estudiantes (nombre apellidos, año académico, provincia, municipio, plan de estudio y grupo) y trabajadores (nombres y apellidos) con el objetivo de asociarlos a un proyecto determinado.

2.15 Conclusiones parciales.

Después de la valoración de cómo se lleva a cabo actualmente y de lo que se pretende lograr una vez que se informaticen, se identificaron requisitos funcionales y no funcionales encargados de viabilizarle el trabajo a los desarrolladores. Con la detección de los mismos se realizará la implementación de una forma más organizada. Se muestra un mapa de navegación que refleja como llegar al módulo Registro y todas sus funcionalidades, para que de esta forma el usuario se sienta orientado una vez que entre a la aplicación.

Capítulo 3: Implementación y prueba

3.1 Introducción

En el siguiente capítulo se definirán los estándares de codificación. Se realizarán pruebas para garantizar el buen funcionamiento y seguridad del sistema, además de manifestar sus resultados.

3.2 Estándar de codificación

Usar técnicas de codificación sólidas y realizar buenas prácticas de programación con vistas a generar un código de alta calidad es de gran importancia para la calidad del *software* y para obtener un buen rendimiento. Además, si se aplica de forma continua un estándar de codificación bien definido caben muchas posibilidades de que un proyecto de *software* se convierta en un sistema fácil de comprender y de mantener.

3.2.1 Denotación, llaves de apertura, cierre y tamaño de líneas

Usar una denotación sin tabulaciones, con un equivalente a 4 espacios, para mantener integridad en las revisiones SVN. El uso de las llaves “{}” será en una nueva línea. La longitud de las líneas de código es aproximadamente de 75-80 caracteres.

Ejemplo:

```
1....$a = $b;  
2  
3....function ejemplo ()  
4.... {  
5....//BI  
6.... }
```

Capítulo 3: Implementación y prueba

3.2.2 Convención de nomenclatura

Variables: se rigen por la nomenclatura camelCase. Siempre comienzan por minúscula y en caso de nombres compuestos la primera letra de cada palabra comienza con mayúscula.

Ejemplo:

```
1....$variable
```

```
2....$variableNombreCompuesto
```

Constantes: siempre deben ser en mayúsculas, con caracteres de subrayado “_” para separar palabras en caso de nombres compuestos.

Ejemplo:

```
1....define (CONSTANTE, valor);
```

```
2....define (CONSTANTE_COMPUESTO, valor);
```

Clases: siempre comienzan con mayúscula, en caso de nombres compuestos las palabras se separan con el carácter subrayado “_” y el resto en minúscula.

Ejemplo:

```
1....class Clase
```

```
2.... {
```

```
3....//BI
```

```
4....}
```

```
5
```

```
6....class Clase_nombre_compuesto
```

Capítulo 3: Implementación y prueba

```
7.... {  
8....//BI  
9....}
```

Funciones: se rigen por la nomenclatura camelCase. Siempre comienzan con minúscula y en caso de nombres compuestos la primera letra de cada palabra comienza con mayúscula. Los parámetros son separados por espacio luego de la coma que los separa.

Ejemplo:

```
1....function funcion ($parametro1,$parametro2)  
2.... {  
3....//BI  
4....}  
5  
6....function funcionNombreCompuesto ($parametro1,$parametro2)  
7.... {  
8....//BI  
9....}
```

Ficheros: todo siempre en minúscula y en caso de nombres compuestos se usa el carácter subrayado “_”.

- Vistas: intuitivo y relacionado con el formulario y/o vista que representa.
- Modelos: con el mismo nombre de la clase que representa que contiene en el nombre el sufijo `_mdl` o `_base` en caso de ser modelos base.

- Librerías: con el mismo nombre de la clase que representa que contiene en el nombre el sufijo `_lib`.
- Controladoras: con el mismo nombre de la clase que representa.

3.2.3 Estructuras de control

Se incluye `if`, `for`, `foreach`, `while` y `switch` entre las estructuras de control. Se recomienda utilizar siempre llaves de apertura y cierre, incluso en situaciones en las que técnicamente son opcionales. Esto aumenta la legibilidad y disminuye la probabilidad de errores lógicos.

Ejemplo:

```
1....if.(condicion)
```

```
2.... {
```

```
3....//BI
```

```
4....}
```

```
5....elseif (condicion)
```

```
6.... {
```

```
7....//BI
```

```
8....}
```

```
9....else
```

```
10.... {
```

```
11....//BI
```

```
12....}
```

```
13
```

```
14....switch.(valor)
```

Capítulo 3: Implementación y prueba

```
15.... {  
16....case valor1:  
17....//BI para valor1  
18....break;  
19....case valor2:  
20....//BI para valor2  
21....break;  
22....default:  
23....//BI por defecto  
24....}
```

Si las condiciones son muy largas que sobrepasan el tamaño de la línea, estas se dividen en varias líneas.

Ejemplo:

```
1....if.(condicion1  
2....|| condicion2)  
3....||(condicion3  
4....&& condicion4))  
5.... {  
6....//BI  
7....}
```

Capítulo 3: Implementación y prueba

En el mejor de los casos cuando la condición es muy extensa, se puede dividir esta en variables y compararlas dentro de la estructura de control.

Ejemplo:

```
1...$variableCondicion1 = condicion1 || condicion2;  
2...$variableCondicion2 = condicion3 &&condicion4;  
3  
4...if ($variableCondicion1 || $variableCondicion2)  
5... {  
6...//BI  
7...}
```

3.2.4 Documentación

Todos los archivos deben tener la documentación asociada al mismo. Para esto debe de cumplir con el siguiente bloque al principio de cada clase.

Clase:

```
1 /**  
2 *Breve descripción de la clase  
3 *  
4 *PHP versión #  
5 *  
6 *@category Categoría de la clase implementada “Librería,  
7 * Controladora, Modelo”
```

Capítulo 3: Implementación y prueba

8 ***@package** Nombre del paquete o módulo al que pertenece

9 ***@author** Nombre y Apellidos del autor y correo electrónico

10*/

Funciones:

1 **/****

2 ***Breve descripción de la función**

3 *****

4 ***@param** tipo y nombre del parámetro(por cada parámetro que

5 *** recibe la función)**

6 ***@return** tipo que retorna

7 ***@author** Nombre y Apellidos del autor y correo electrónico

8 ***/**

3.2.5 Buenas prácticas

Los valores booleanos y nulos siempre se escriben con mayúscula, para facilitar la legibilidad del código, usar un “enter” antes de las estructuras de control y definición de las funciones.

1....**\$variableBooleana = FALSE;**

2....**\$variableNula = NULL;**

3....

4....**if (condicion)**

5.... **{**

6....**//BI**

3.3. Validación de los Requisitos Funcionales

La validación de requisitos tiene como misión demostrar que la definición de los requisitos define realmente el sistema que el usuario necesita o el cliente desea. La validación de requisitos es el proceso de verificarlos en cuanto a validez, consistencia, integridad, realismo y certidumbre.

3.3.1 Técnicas de validación

Sommerville define un grupo de técnicas que permiten que el proceso de validación tenga una mejor calidad, de ellas se utilizaron:

Construcción de prototipos: se mostró un modelo ejecutable del sistema a los clientes para que los mismos interactuaran con este y determinaran si cumplía o no con sus necesidades.

Generación de casos de pruebas: se realizaron los diseños de casos de pruebas para cada uno de los requisitos anteriormente planteados, permitiendo verificar que todos se pudieran probar, e identificar dependiendo de la complejidad del diseño de caso de prueba.

3.4 Pruebas

La prueba es el proceso de ejecución de un programa con la intención de descubrir un error. El objetivo es diseñar pruebas que sistemáticamente saquen a la luz diferentes clases de errores, haciéndolo con la menor cantidad de tiempo y de esfuerzo. Si la prueba se lleva a cabo con éxito, descubrirá errores en el *software*. Pero, la prueba no puede asegurar la ausencia de defectos; solo puede demostrar que existen defectos en el *software* [28] .

3.4.1 Niveles de pruebas

Para la evaluación del *software* se debe comenzar por los componentes más pequeños hasta haber logrado probar el mismo en su totalidad. En el trabajo se siguen los siguientes pasos que ayudan a que este se realice de manera secuencial:

Capítulo 3: Implementación y prueba

- Pruebas de unidad
- Pruebas del sistema
- Pruebas de aceptación

Pruebas unitarias

La prueba de unidad se concentra en el esfuerzo de verificación de la unidad más pequeña del diseño del *software*: el componente o módulo de *software*. Tomando como guía la descripción del diseño al nivel de componentes, se prueban importantes caminos de control para descubrir errores dentro de los límites del módulo. Estas pruebas se concentran en la lógica del procesamiento interno y en las estructuras de datos dentro de los límites de un componente [29] .

Pruebas del sistema

Este tipo de prueba está constituida por una serie de pruebas diferentes cuyo propósito primordial es ejercitar profundamente el sistema para verificar que se han integrado adecuadamente todos los elementos del mismo (*hardware*, otro *software*, etcétera) y que realizan las funciones adecuadas [30] .

Pruebas de aceptación

La prueba de aceptación del usuario es la última acción de prueba antes de desplegar el *software*. El objetivo de la misma es comprobar si el *software* está preparado y lo pueden utilizar los usuarios para realizar las funciones y tareas para las que se diseñó. Una prueba de aceptación con éxito en esta etapa puede ser un requerimiento contractual en la primera fase de ser firmado por el cliente. Como resultado de esta prueba se obtuvo la plantilla de aceptación del cliente la cual fue aceptada y firmada satisfactoriamente por los mismos.

3.4.2 Métodos y técnicas de prueba

Los métodos de pruebas se agrupan en :

- **Métodos de caja blanca o estructurales:** se llevan a cabo sobre la interfaz del *software*. Con los diseños de casos de prueba pretenden demostrar que las funciones del *software* son operativas, que la entrada se acepte de forma adecuada y se produce un resultado correcto.

Capítulo 3: Implementación y prueba

- **Métodos de caja negra o funcionales:** se llevan a cabo sobre la interfaz del *software*. Con los diseños de casos de prueba pretenden demostrar que las funciones del *software* son operativas, que la entrada se acepte de forma adecuada y se produce un resultado correcto. No es necesario conocer la lógica del mismo, únicamente la funcionalidad que debe realizar.

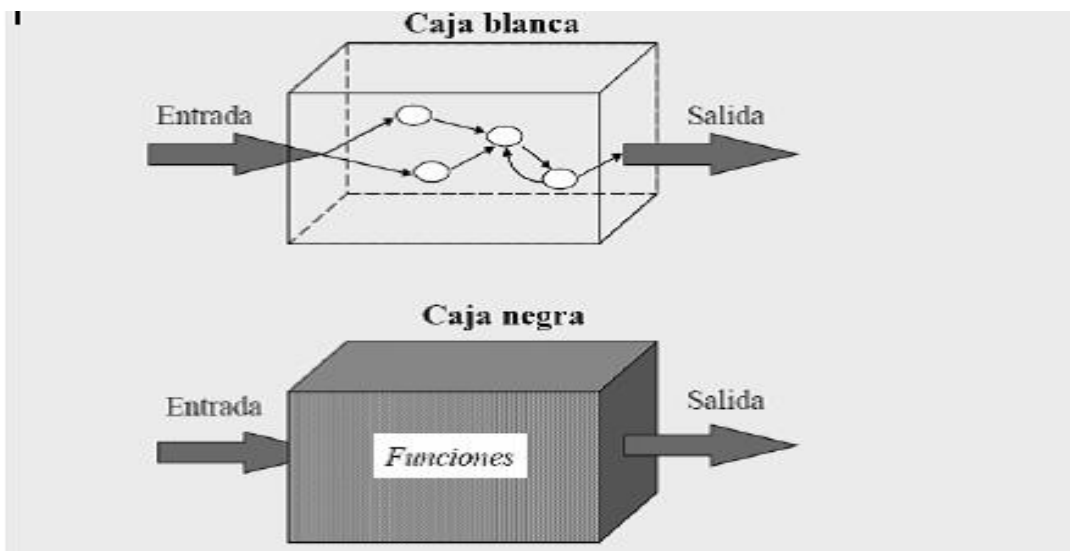


Figura 7: técnicas de caja negra y caja blanca

Técnicas de caja blanca

De las técnicas de caja blanca, en el trabajo se tuvo en cuenta la de prueba del camino básico, la cual se especifica a continuación:

Prueba del camino básico: esta técnica permite obtener una medida de la complejidad lógica de un diseño y usar la misma como guía para la definición de un conjunto básico. La idea es derivar casos de prueba a partir de un conjunto dado de caminos independientes por los cuales puede circular el flujo de control. Para obtener dicho conjunto de caminos independientes se construye el grafo de flujo asociado y se calcula su complejidad ciclomática.

El algoritmo a analizar es el que se presenta a continuación:

Capítulo 3: Implementación y prueba

```
<?php
public function registrar() {
    $post_vars = $this->input->all_post();           1
    $datos = $this->input->all_post(true);           1
    $conceptos_asociados = json_decode($datos['datosGrid']); 1
    $conceptos_asociados1 = json_decode($datos['datosGrid1']); 1
    $datos0 = array();$datos1 = array();$datos2 = array();$datos3 = array(); 1|
    $datos3['id_persona'] = $post_vars['idPersona'];         1
    $datos3['id_anno_academico'] = $post_vars['idAnno'];     1
    $datos0['id_persona'] = $post_vars['idPersona'];         1
    $datos0['id_anno_academico'] = $post_vars['idAnno'];     1
        if (!empty($conceptos_asociados)) { 1
            $this->extension_lib->registrarEventosDeportivos($datos, $conceptos_asociados, $datos0); 2
        }
        if (!empty($conceptos_asociados1)) { 3
            $this->extension_lib->registrarEventosCulturales($datos, $conceptos_asociados1, $datos0); 4
        }
        if (!empty($post_vars['residencia'])) { 5
            $datos1['id_persona'] = $post_vars['idPersona']; 6
            $datos1['id_anno_academico'] = $post_vars['idAnno']; 6
            $datos1['id_evaluacion'] = $post_vars['residencia']; 6
            $this->extension_lib->registrarEvaluacionResidencia($datos1); 6
        }
        if (!empty($post_vars['cuarteleria'])) { 7
            $datos2['id_evaluacion'] = $post_vars['cuarteleria']; 8
            $this->extension_lib->registrarEvaluacionCuarteleria($datos2); 8
        }
        if (!empty($post_vars['guardia'])) { 9
            $datos3['id_evaluacion'] = $post_vars['guardia']; 10
            $this->extension_lib->registrarEvaluacionGuardia($datos3); 10
        }
    } 11
?>
```

Figura 8: código de registrar actividades de extensión

Después de este paso, es necesario representar el grafo de flujo asociado, en el cual se representan distintos componentes como es el caso de:

Nodo: son los círculos representados en el grafo de flujo, el cual representa una o más secuencias del procedimiento, donde un nodo corresponde a una secuencia de procesos o a una sentencia de decisión. Los nodos que no están asociados se utilizan al inicio y final del grafo.

Capítulo 3: Implementación y prueba

Aristas: son constituidas por las flechas del grafo, son iguales a las representadas en un diagrama de flujo y constituyen el flujo de control del procedimiento. Las aristas terminan en un nodo, aun cuando el nodo no representa la sentencia de un procedimiento.

Regiones: son las áreas delimitadas por las aristas y nodos donde se incluye el área exterior del grafo, como una región más. Las regiones se enumeran siendo la cantidad de regiones equivalente a la cantidad de caminos independientes del conjunto básico de un procedimiento.

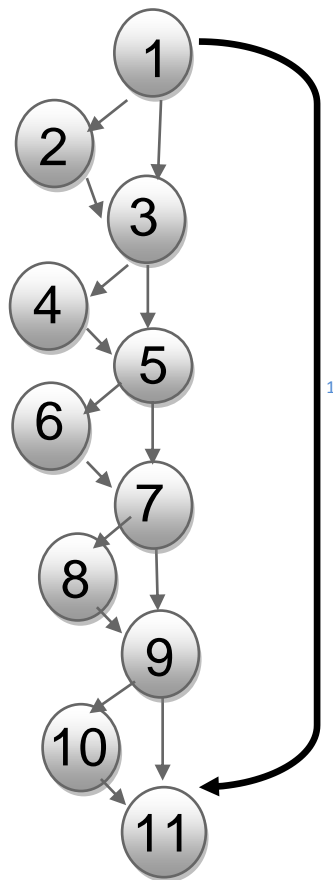


Figura 9: grafo de los caminos del método

Fórmula para calcular la complejidad ciclomática:

$$V(G) = (A - N) + 2$$

$$V(G) = (15 - 11) + 2$$

$$V(G) = 6$$

Capítulo 3: Implementación y prueba

Siendo “A” la cantidad total de aristas y “N” la cantidad total de nodos.

En el cálculo efectuado la complejidad ciclomática del código es de 6, lo que significa que existen a lo sumo seis posibles caminos por donde el flujo puede circular, este valor representa el límite mínimo del número total de casos de pruebas para el procedimiento tratado.

Seguidamente es necesario representar los caminos básicos por los que puede recorrer el flujo.

Camino básico No. 1: 1,3,5,7,9,11

Camino básico No. 4: 1,2,3,4,5,6,7,9,11

Camino básico No. 2: 1,2,3,5,7,9,11

Camino básico No. 5: 1,2,3,4,5,6,7,8,9,11

Camino básico No. 3: 1,2,3,4,5,7,9,11

Camino básico No. 6: 1,2,3,4,5,6,7,8,9,10,11

Después de haber extraído los caminos básicos del flujo, se procede a ejecutar los casos de pruebas para este procedimiento, se debe realizar al menos un caso de prueba por cada camino básico. Para realizar los casos de pruebas es necesario cumplir con las siguientes exigencias:

Casos de prueba

Camino 1: [1,3,5,7,9,11]

Descripción: todos los campos estarán vacíos.

Entrada: un arreglo vacío.

Resultados esperados: se muestra en mensaje de error.

Camino 2: [1,2,3,5,7,9,11]

Descripción: todos los campos estarán vacíos menos los campos de eventos deportivos.

Entrada: un arreglo que tiene datos relacionados con los eventos deportivos.

Resultados esperados: el sistema registra los datos relacionados con los eventos deportivos.

Camino 3: [1,2,3,4,5,7,9,11]

Descripción: todos los campos estarán vacíos menos los campos de eventos culturales.

Capítulo 3: Implementación y prueba

Entrada: un arreglo que tiene datos relacionados con los eventos culturales.

Resultados esperados: el sistema registra los datos relacionados con los eventos culturales.

Camino 4: [1,2,3,4,5,7,9,11]

Descripción: todos los campos estarán vacíos menos los campos de residencia.

Entrada: un arreglo que tiene datos relacionados con residencia.

Resultados esperados: el sistema registra los datos relacionados con residencia.

Camino 5: [1,2,3,4,5,6,7,8,9,11]

Descripción: todos los campos estarán vacíos menos los campos de cuartería.

Entrada: un arreglo que tiene datos relacionados con cuartería.

Resultados esperados: el sistema registra los datos relacionados con cuartería.

Camino 6: [1,2,3,4,5,6,7,8,9,10,11]

Descripción: todos los campos estarán vacíos menos los campos de guardia.

Entrada: un arreglo que tiene datos relacionados con guardia.

Resultados esperados: el sistema registra los datos relacionados con guardia.

Técnicas de caja negra

En el trabajo se aplican los diseños de casos de pruebas como técnicas de caja negra. A continuación se explica en qué consisten estos y se muestra un caso de prueba para un requisito funcional.

Casos de pruebas: un caso de prueba es una especificación, usualmente formal, de un conjunto de, entradas de prueba condiciones de ejecución y resultados esperados, identificados con el propósito de hacer una evaluación de aspectos particulares de un elemento objeto de prueba.

Capítulo 3: Implementación y prueba

Escenario	Descripción	Respuesta del sistema	Flujo central
Mostrar detalles de Metodología.	Mediante este escenario se pueden mostrar detalles de metodología.	El sistema muestra un mensaje donde aparece: "Metodología". "Descripción". "Estado".	<ul style="list-style-type: none"> • El usuario una vez autenticado en el sistema selecciona el subsistema "CICE" el cual se encuentra en el escritorio. • El usuario selecciona el módulo "Registro" del subsistema "CICE". • El sistema muestra diferentes opciones del menú. • El usuario selecciona el elemento "Configuración" y luego la opción "Metodologías", el sistema muestra un listado de las metodologías introducidos hasta la fecha. • El usuario selecciona la opción "Detalles" en el área de iconos internos del listado de metodologías y se muestran los datos en forma de ventana emergente.

Tabla 5: Mostrar detalles de metodologías.

Luego de haberle realizado casos de pruebas al módulo Registro, se detectaron los siguientes resultados en cada una de las iteraciones:

- En la primera iteración se detectaron 40 no conformidades, encontrándose errores en los íconos internos, estos no realizaban la acción que les correspondía, además de encontrarse errores a la hora de mostrar los mensajes y en las funcionalidades de registrar. Todas estas no conformidades fueron corregidas.
- En la segunda se detectaron 15, no se mostraban todas las interfaces y los filtros de búsqueda tenían algunos errores a la hora de buscar. Todo fue corregido.
- En la tercera iteración se detectaron 2 no conformidades, encontrándose errores ortográficos y siendo corregidos.

- En la cuarta iteración no se detectaron errores, por lo que se cumplió con los requisitos funcionales anteriormente planteados para darle solución a la problemática.

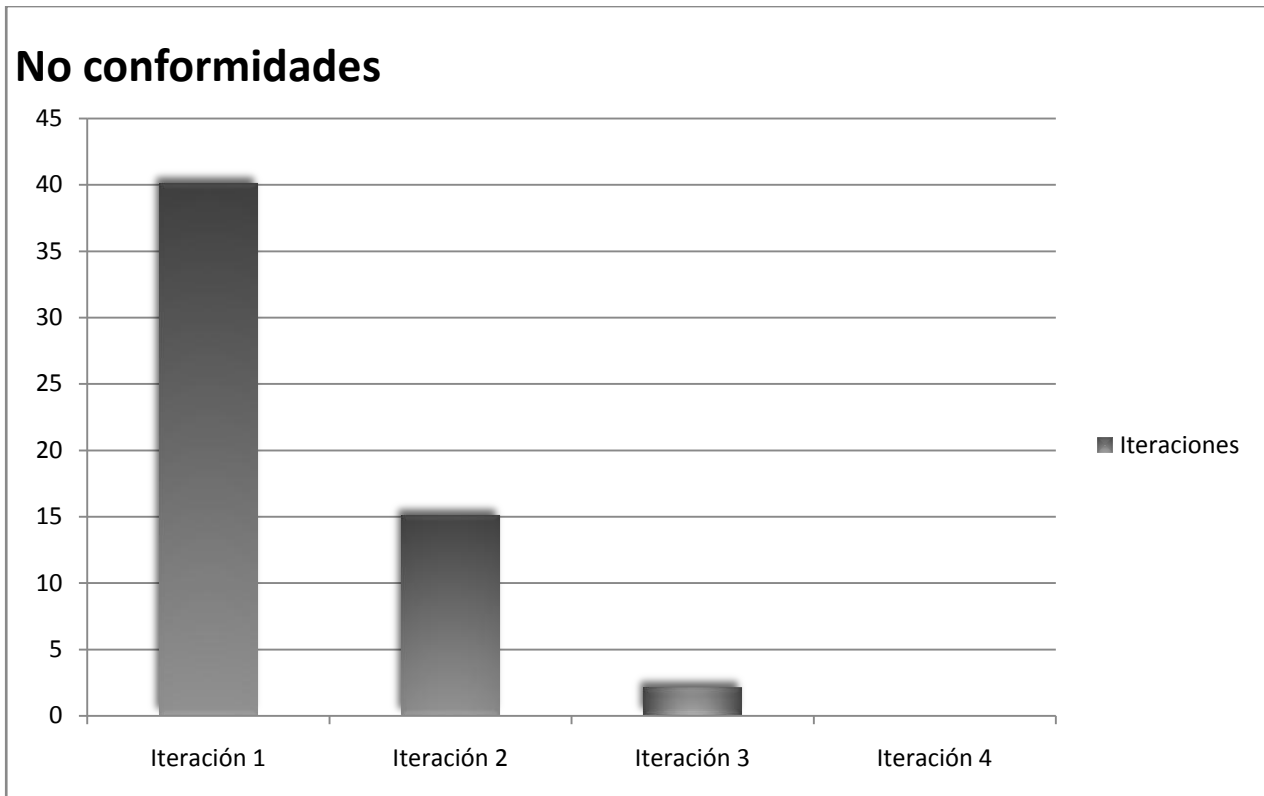


Figura 10: no conformidades

Dentro de los métodos de caja negra se encuentra el tipo de prueba de stress, a continuación se explicará brevemente en que consiste y el resultado obtenido después de aplicársela a la aplicación utilizando la herramienta **JMeter 2.6**.

Pruebas de resistencia o de stress: están diseñadas para enfrentar a los programas con situaciones anormales. Este tipo de pruebas ejecuta un sistema de forma que demande recursos en cantidad, frecuencia o volúmenes anormales hasta que el sistema falla. Entre sus principales funciones están [30] .

- Prueba el comportamiento de fallo de ejecución del sistema.
- Manifestar defectos que normalmente no serían descubiertos.

Capítulo 3: Implementación y prueba

Etiqueta	# Muestras	Media	Mediana	Línea de 90%	Mín	Máx	% Error	Rendimiento	Kb/sec
Registrar de actividades de f...	100	20999	20999	21003	20997	21026	0,00%	4,5/sec	7,5
Registro de actividades de e...	100	20999	20999	21000	20997	21004	0,00%	4,5/sec	7,5
Registro de medidas disciplin...	100	20999	20999	21000	20997	21007	0,00%	4,5/sec	7,5
Registro de competencias té...	100	20999	20999	21000	20997	21003	0,00%	4,5/sec	7,5
Total	400	20999	20999	21001	20997	21026	0,00%	4,7/sec	7,8

Figura 11: resultados de pruebas del JMeter

Después de la aplicación de esta prueba con la utilización de la herramienta JMeter, se ofrecen datos de interés sobre el stress que puede soportar el sistema. Para 100 usuarios conectados en 10 hilos de manera concurrente a 4 de las principales funcionalidades, se obtuvo como resultado el correcto funcionamiento del mismo, no presentando errores al ser utilizado por los usuarios.

3.5 Conclusiones parciales.

En el capítulo se realizó la descripción de las pruebas a las que fue sometida la aplicación, con la intención de verificar el buen funcionamiento de la misma. Se concluye que la definición de los requisitos funcionales permitieron facilitar la implementación de las funcionalidades, con la ayuda de las buenas prácticas utilizadas, cumpliendo de esta manera los objetivos trazados y obteniendo un sistema que permita el registro de información de las actividades realizadas por los estudiantes de la UCI.

Conclusiones

Con la finalización del trabajo realizado, se ha confirmado que es verdaderamente importante y necesaria la implementación del módulo Registro, pues este permitió la informatización de toda la información del estudiante en las diferentes esferas que evalúa la universidad, además de que se resolvió de forma eficiente uno de los problemas que afecta a la UCI en la actualidad, el registro de información para las caracterizaciones de los estudiantes, de manera general se concluye con lo siguiente:

- Se estudiaron de forma satisfactoria, sistemas informáticos vinculados con la gestión de la información identificando ventajas y deficiencias de los mismos, permitiendo erradicar los problemas de los sistemas existentes y fusionar las mejores prácticas y funcionalidades.
- Se utilizaron para el correcto funcionamiento de este módulo las metodologías, herramientas y lenguajes acordes a las necesidades del país de migrar al *software* libre.
- Se realizó un estudio de los estándares de codificación elegidos por la línea de arquitectura, encargada de seleccionar todo lo relacionado con la arquitectura del proyecto, lo cual permitió entender el por qué de su selección, siendo de gran utilidad en la implementación; ya que sirvió para aprender nuevos métodos y formas de tener organizado el código.
- El empleo de un proceso de desarrollo con enfoque ágil permitió el trabajo en equipo en conjunto con el cliente, además del desarrollo de la aplicación de una forma más rápida y centrada en la solución del problema identificado
- Se cumplió el objetivo de la investigación, pues se realizó la implementación del módulo Registro de información para las caracterizaciones de los estudiantes de la UCI, cumpliendo con todos los requisitos establecidos.
- Se realizaron diferentes tipos de pruebas con el objetivo de comprobar la viabilidad de la solución.

El sistema resultante logró cubrir todas las funcionalidades previstas, favoreciendo la organización, veracidad e integridad de la información, brindando una gran confidencialidad del proceso de registro de la información para las caracterizaciones de los estudiantes de la UCI.

Recomendaciones

Con el estudio realizado y teniendo en cuenta el proceso de mejora continua del desarrollo de *software* se realizan las siguientes recomendaciones:

- Realizar otra versión del producto, la cual sea aplicable no solo para la UCI, sino para todas las enseñanzas educacionales del país.
- Brindar la posibilidad de realizar las funcionalidades de exportar e imprimir la información referente al estudiante.

Referencias bibliográficas

- [1] Redmine. [En línea] UCI, 2010. [Citado el: 11 de junio de 2012.] Disponible en: <http://gespro.cenia.prod.uci.cu/projects/plantillacenib>.
- [2] Dirección de información. [En línea] [Citado el: 11 de abril de 2012.] Disponible en: <http://biblioteca2.uci.cu/>.
- [3] **Lic. Belina Capote Marrero.** La gestión de información como herramienta fundamental en el desarrollo de los centros toxicológicos . [En línea] [Citado el: 11 de abril de 2012.] Disponible en: http://www.bvs.sld.cu/revistas/aci/vol11_2_03/aci030203.htm.
- [4] definición.org. [En línea] [Citado el: 11 de abril de 2012.] Disponible en: <http://www.definicion.org/informacion>.
- [5] monografías.com. [En línea] [Citado el: 15 de mayo de 2012.] Disponible en: (<http://www.monografias.com/trabajos91/sistemas-informacion-proyeccion-al-exito/sistemas-informacion-proyeccion-al-exito.shtml>).
- [6] slideshare. [En línea] [Citado el: 11 de mayo de 2012.] Disponible en: <http://www.slideshare.net/rutatec/sistema-de-gestin-escolar-1732387>.
- [7] DE beta. [En línea] [Citado el: 13 de mayo de 2012.] Disponible en: <http://www.de.gobierno.pr/informacion-para-padres-sobre-sistema-de-informacion-estudiantil-sie>.
- [8] monografía.com. [En línea] [Citado el: 13 de mayo de 2012.] Disponible en: <http://www.monografias.com/trabajos60/sistema-informativo-estudiantil/sistema-informativo-estudiantil.shtml>.
- [9] [En línea] [Citado el: mayo de 5 de 2012.] agilemanifesto.org/iso/es/principles.html.
- [10] Calisoft centro de la calidad . [En línea] [Citado el: mayo de 5 de 2012.] Disponible en: <http://calisoft.uci.cu/index.php/proceso-de-mejora>.
- [11] [En línea] [Citado el: 10 de mayo de 2012.] Disponible en: <http://es.scribd.com/doc/90454268/Act-a-Uml>.
- [12] [En línea] [Citado el: 25 de abril de 2012.] Disponible en: <http://www.pencil-animation.org/>.
- [13] The Apache Software Foundation. Apache JMeter - Apache JMeter™ . [En línea] [Citado el: 11 de abril de 2012.] Disponible en: <http://jmeter.apache.org>.
- [14] ELLISLAB INC. *CodeIgniter* [en línea]. 2011 [fecha de consulta: 19 Octubre 2011] Disponible en: <http://www.codeigniter.com>

Referencias bibliográficas

- [15] THE JQUERY FOUNDATION. *JQuery* [en línea]. 2011. [fecha de consulta: 20 Octubre 2011] Disponible en: <http://jquery.com>
- [16] ADRI MARTIN, Sergio. *PostgreSQL* [en línea]. Madrid: Editorial Academia Española, 2011 [fecha de consulta: 20 Octubre 2011]. Disponible en: http://books.google.com.cu/books?id=NmB_tQAACAAJ&dq=postgreSql&hl=es&sa=X&ei=97e2T9PSIsKjg wfF_e1H&ved=0CFMQ6AEwBAISBN: 9783846575277
- [17] PGADMIN. *Introduction* [en línea]. 2011 [fecha de consulta: 6 Noviembre 2011]. Disponible en: <http://www.pgadmin.org>
- [18] J. KABIR, Mohammed. *Servidor Apache 2: la biblia* [en línea]. [s.l.]: Anaya Multimedia, 2009 [fecha de consulta: 27 Octubre 2011]. Disponible en: <http://sunshine.prod.uci.cu/search/apache>
- [19] ORACLE Corporation. *NetBeans* [en línea]. 2011 [fecha de consulta: 4 Noviembre 2011] Disponible en: http://netbeans.org/index_es.html
- [20] VAN LANCKER, Luc. *CSS 1 y CSS 2.1: Hojas de estilo para enriquecer el código HTML* [en línea]. Barcelona: Ediciones ENI, 2007 [fecha de consulta: 22 Octubre 2011]. Disponible en: http://books.google.com.cu/books?id=xBrgWWa31pcC&printsec=frontcover&dq=css&hl=es&sa=X&ei=n7WzT_SCDliXtweouY3NCA&ved=0CGYQ6AEwCQ#v=onepage&q=css&f=false ISBN: 9782746035836
- [21] SÁNCHEZ MAZA, Miguel Ángel. *Javascript* [en línea]. [s.l.]: Editorial Innovación y Cualificación, 2006 [fecha de consulta: 20 Octubre 2011]. Disponible en: http://books.google.com.cu/books?id=3x09sewjaHIC&printsec=frontcover&dq=javascript&hl=es&ei=wpC5TsrED62DsgLs9ZSjCA&sa=X&oi=book_result&ct=result&resnum=1&ved=0CCwQ6AEwAA#v=onepage&q&f=false. ISBN: 8495733188.
- [22] Scribd. [En línea] [Citado el: 10 de abril de 2012.] <http://es.scribd.com/doc/94998619/41/Sistema-administrador-de-base-de-datos-relacionales>.
- [23] Tecnología y Synergix. [En línea] [Citado el: 10 de junio de 2012.] <http://synergix.wordpress.com/2008/07/10/modelo-de-dominio/>.
- [24] **Pressman**. *Ingeniería del Software, Un Enfoque Práctico*. [aut. libro] Pressman y Roger S. Pressman. 2007.
- [25] SOMMERVILLE, Ian. *Ingeniería del software* [en línea]. 7ma Edición [s.l.]: Pearson Educación, 2005 [fecha de consulta: 2 Enero 2012]
- [26] EcuRed. [En línea] [Citado el: 15 de abril de 2012.] http://www.ecured.cu/index.php/Patr%C3%B3n_Modelo_Vista_Controlador.

Referencias bibliográficas

[27] Scribd. [En línea] [Citado el: 23 de mayo de 2012.] <http://es.scribd.com/doc/27239031/PATRONES-DE-DISENO>.

[28] **Pressman**. *Ingeniería del Software, Un enfoque ágil. Ingeniería del Software, Un enfoque ágil*. quinta edición.

[29] **Roger pressman** . *Ingeniería del software. Un enfoque práctico, 6ta edición. Cap 13*.

[30] PRESSMAN, Roger S. *Ingeniería de Software: Un enfoque práctico*. 7ma. Edición. New York: McGraw-Hill Companies, 2010 [fecha de consulta: 5 Enero 2012]. Disponible en: http://eva.uci.cu/mod/resource/view.php?id=8500&subdir=/Ediciones_del_Pressman/Pressman_7ma_edicion. ISBN: 9780073375977.

Bibliografía consultada

- [1] Vates, 2012. [Disponible en: <http://www.vates.com/cmami/cmami.html>]
- [2] W3C, 2012. [Disponible en: <http://www.w3c.es/Consortio>]
- [3] Desarrolloweb.com. [Disponible en: <http://www.desarrolloweb.com/articulos/1696.php>]
- [4] Completa el paquete de desarrollo Java, 2012. [Disponible en: <http://translate.google.com/cu/translate?hl=es&sl=en&u=http://www.filehorse.com/download-netbeans/5372/&ei=nOm8T6L9DsztggeUvriPAQ&sa=X&oi=translate&ct=result&resnum=10&ved=0CHMQ7gEwCQ&prev=/search%3Fq%3Dque%2Bes%2Bel%2Bnetbeans%2B6.7.1%26hl%3Des%26biw%3D939%26bih%3D548%26prmd%3Dimvnsa>]
- [5] NetBeans, 2012.[Disponible en: <http://netbeans.org/community/releases/67/>]
- [6] Boost Productivity with Innovative and Intuitive Technologies. [Disponible en: <http://www.visual-paradigm.com>]
- [7] el altareao. [Disponible en: <http://www.atareao.es/ubuntu/software-para-tu-ubuntu/evolus-pencil-una-forma-sencilla-de-disenar-aplicaciones>]
- [8] De todo, 2010. [Disponible en: <http://sudokill.blogspot.com/2010/10/evolus-pencil.html>]
- [9] HTMLPOINT.com, 1997-2006. [Disponible en: http://www.htmlpoint.com/javascript/corso/js_02.htm]
- [10] Patrón “Modelo-Vista-Controlador”. [Disponible en: www.proactivacalidad.com/java/patrones/mvc.html]
- [11] neleste2.0, 2012. [Disponible en: www.neleste.com/modelo-vista-controlador/]
- [12] Diccionarios.com, 2011. [Disponible en: <http://www.diccionarios.com>]
- [13] desarrolloweb.com. [Disponible en: <http://www.desarrolloweb.com/articulos/desarrollo-agil.html>]
- [14] (Joaquín García). IngenieroSoftware, 2003. [Disponible en: <http://www.ingenierosoftware.com/calidad/cmm-cmami-nivel-2.php>]
- [15] Desarrolloweb.com. [Disponible en: www.desarrolloweb.com/manuales/58/]
- [16] (Sommerville). Ingeniería del Software, 7ma edición, 2005.
- [17] Guglielmetti, 2004
- [18] Bases y principios del proceso de enseñanza-aprendizaje centrado en el aprendizaje en la UCI.